Engineering Governance

Version 1.0

6 December 2007

Robert W. Stavros, PhD

MITRE

Michael B. Dettman

Associate Technical Director (Policy and Guidance), PEO C4I

Jeremiah Albrant

Space and Naval Warfare Systems Center, San Diego

CONTENTS

List of Figures

List of Tables

ABSTRACT

The goal of Engineering Governance is to define a general model that describes the various aspects of governance that can be the basis of communication among governance efforts. Three aspects (Regulation, Execution and Compliance) comprise Governance. Combining these aspects with the five layers (Data, Information, Knowledge, Understanding and Wisdom) of the Cognitive Model results in 15 different governance roles. Aspect-specific Conceptual Data Models describe each of the roles. Governance objects and the relationships between the objects comprise the Conceptual Models. Finally, there are behavior rules for some of the objects.

ENGINEERING GOVERNANCE

As Information Technology (IT) has grown and matured, higher level human concepts are now subject to the same pressures for automation and digital processing as many other features of our lives. One such concept is Governance which has become a topic *de jour*. Table 1 illustrates this point.

Table 1. Google Results from 24 September 2007

Results	Topic
71,300,00	governance
61,900,00	corporate governance
2,120,00	good governance
1,990,00	SOA governance
1,770,00	IT governance
581,00	data governance

There is an abundance of information about governance on the Internet; however, there is little formalized information on what governance is, or how to model governance.

> *Although the need for governance has long been acknowledged in practice and academia, papers discussing an actual governance model are scarce.*
> [IEEE, Gewald H. Helbig]

The first step in developing a Governance Model is to decompose governance into smaller, more manageable pieces that lend themselves to modeling and ultimately automation. Modeling depends on the creation of abstract models. A working definition of an abstract model follows:

> *An abstract model (or conceptual model) is a theoretical construct that represents something with a set of variables and a set of logical and quantitative relationships between them. Models in this sense are constructed to enable reasoning within an idealized logical framework about these processes and are an important component of scientific theories.*
> [Adapted from *Wikipedia: Model (Abstract)*; accessed 9 July 2007]

The following two abstract models used together form the basis of a higher level Governance Conceptual Model. Use this model to perform the engineering analysis necessary to implement a Governance Model successfully.

- Fundamental Governance Model
- Cognitive Model

1.1 Fundamental Governance Model

Governance is not a synonym for government or for regulations; rather, governance is the process governments use to interpret and use regulations.

> *Governance is that separate process or certain part of management or leadership processes that make decisions that define expectations, grant power, or verify performance. Frequently a government is established to administer these processes and systems.*
> [Adapted from Wikipedia: Governance; accessed 9 July 2007]

This definition offers three aspects as to what comprises governance:

- Making decisions that define expectations
- Granting power
- Verifying performance

The first aspect of governance conveys **Regulation**, the second aspect conveys **Execution,** and the third aspect conveys **Compliance**, as represented in the following model.

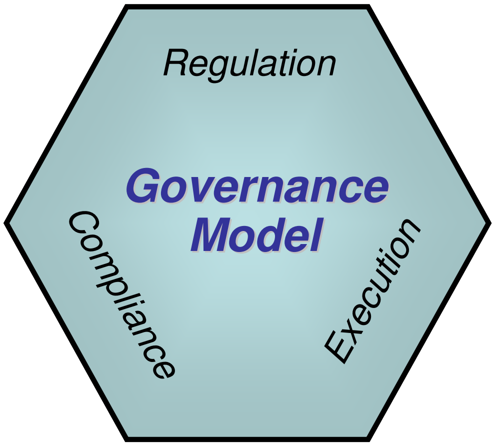


Figure 1. General Governance Model

Good governance is a balance of all three Governance Model aspects; it is meaningless to have Regulation without Execution or Execution without Compliance. In other words, Regulation indicates what needs to be done, Execution is actually doing it, and Compliance is making sure it is done correctly.

1.1.1 Regulation

Regulations are formal, codified, authoritative rules. They are adopted by a public regulatory agency and usually are interpretations of the statutes passed by a legislative body.

> *A regulation as a legal term is a rule created by an administration or administrative agency or body that interprets the statutes setting out the agency's purpose and powers, or the circumstances of applying the statute. A regulation is a form of secondary legislation which is used to implement a primary piece of legislation appropriately, or to take account of particular circumstances or factors emerging during the gradual implementation of, or during the period of, a primary piece of legislation.*
> [Adapted from Wikipedia: Regulation; accessed 9 July 2007]

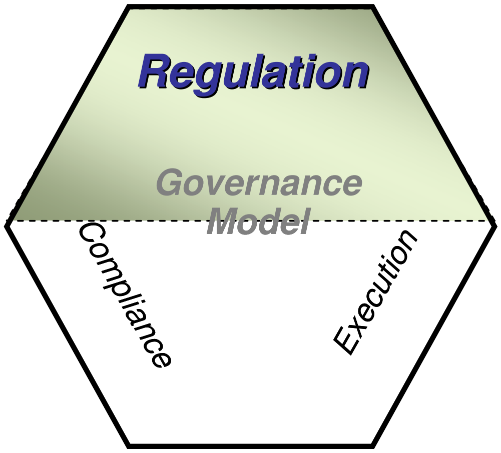


Figure 2. Regulation Aspect of Governance Model

An example of the Regulation aspect of Governance is a body that creates statutes such as the U.S. Congress or a state legislature, an agency that creates or enforces statutes such as the U.S. Internal Revenue Service (IRS), or civilian or commercial groups that create and promote standards such as the Object Management Group (OMG).

1.1.2 Execution

Execution is the aspect of Governance charged with actually fulfilling formal, codified authoritative rules derived from regulation to those specifications provided by compliance. The responsibility for executing the regulation rarely, if ever, falls on the legislative body or those responsible for enforcing the compliance to the regulation. Without Execution, the other aspects of the Governance Model are meaningless. Consequently, any discussion of Governance must include the Execution aspect.

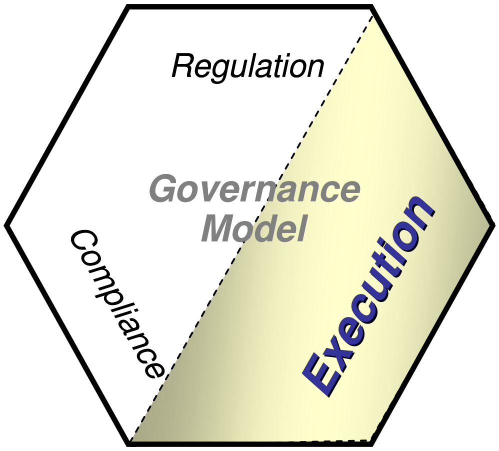


Figure 3. Execution Aspect of Governance Model

Examples of the Execution aspect are the individuals that file their tax forms to the IRS and personnel who actually create the functionality needed by a DoD Program, Project or Initiative.

1.1.3 Compliance

Compliance ensures the objective and valid meeting of regulations through observation, measurement or testing. Good governance cleanly and effectively separates the responsibility for creating regulations from the enforcement of regulations. This does not mean that regulation can be developed in a vacuum; it must be written to be enforceable through compliance checking. Therefore, the line between regulation and compliance is not fixed and rigid but needs to be negotiated with validation of regulations from the Compliance aspect.

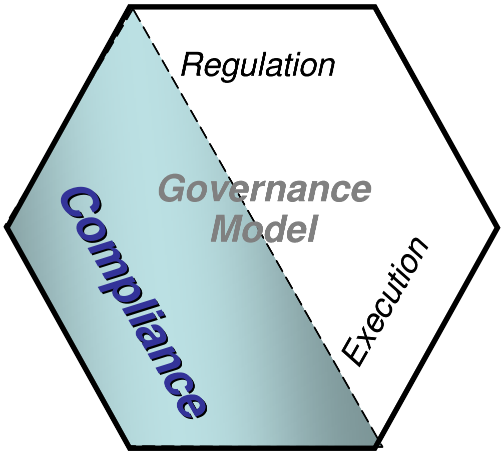


Figure 4. Compliance Aspect of Governance Model

Examples of the Compliance aspect of Governance include the auditing functionality of the IRS and the independent verification and validation (IV&V) functionality within the DoD.

1.2 Cognitive Model

The Cognitive Model abstractly represents human cognition which the American Heritage Dictionary defines as follows:

1. *The mental process of knowing, including aspects such as awareness, perception, reasoning, and judgment.*
2. *That which comes to be known, as through perception, reasoning, or intuition; knowledge.*

Cognition roughly maps to the Information Science and Knowledge Management *DIKW* (*Data*, *Information*, *Knowledge* and *Wisdom*) hierarchies. See The Origin of the “Data Information Knowledge Wisdom” Hierarchy, Nikhil Sharma, 2005 for a complete discussion of DIKW.

Table 2. Mapping Cognitive Aspects to DIKW Hierarchy

Cognitive Aspects	DIKW Hierarchy
Awareness	Data
Perception	Information
Reasoning	Knowledge
Judgment	Wisdom

In addition to the basic layers in the DIKW Hierarchy, Russell Ackoff and Milan Zeleny propose an additional layer between *Knowledge* and *Wisdom*. Ackoff refers to it as *Understanding*. Zeleny adds one more layer above Wisdom called *Enlightenment*. For the purposes of governance, there does seem to be a need for an *Understanding* layer to the hierarchy. However, adding an *Enlightenment* layer when referring to governance always seems to elicit smiles.

The result is termed the *Cognitive Model* instead of the DIKW (or DIKUW) Hierarchy for several reasons. The word *hierarchy* implies an order or precedence and this hierarchy always starts with data. This is a useful concept when thinking in terms of Information Science and Knowledge Management which generally try to organize and classify large amounts of data and extract wisdom or in Zeleny’s case even enlightenment. In governance the hierarchy is applicable in both directions (i.e., from *Wisdom* to *Data* and from *Data* to *Wisdom*).

Another problem with the hierarchical approach is that although the relationship of data to wisdom in some cases is many-to-one (i.e., many pieces of data contribute to a single piece of wisdom), the reality is that relationship is more of a network where one piece of data may ultimately be part of many pieces of wisdom.

The acronym DIKW (or DIKUW, etc.) is not very pronounceable and the term specifically captures the model as we currently understand it. Consequently, as our understanding of the model evolves, as with the acceptance of having an *Understanding* layer, the name of the model must also change.

The Cognitive Model is below. It has the five layers of the original DIKUW Hierarchy, and this view is from *Wisdom* to *Data*. However, the model could just as well be presented in the reverse starting with *Data* and ending in *Wisdom*. The direction through the model is inconsequential and reflects the higher level human cognitive process.

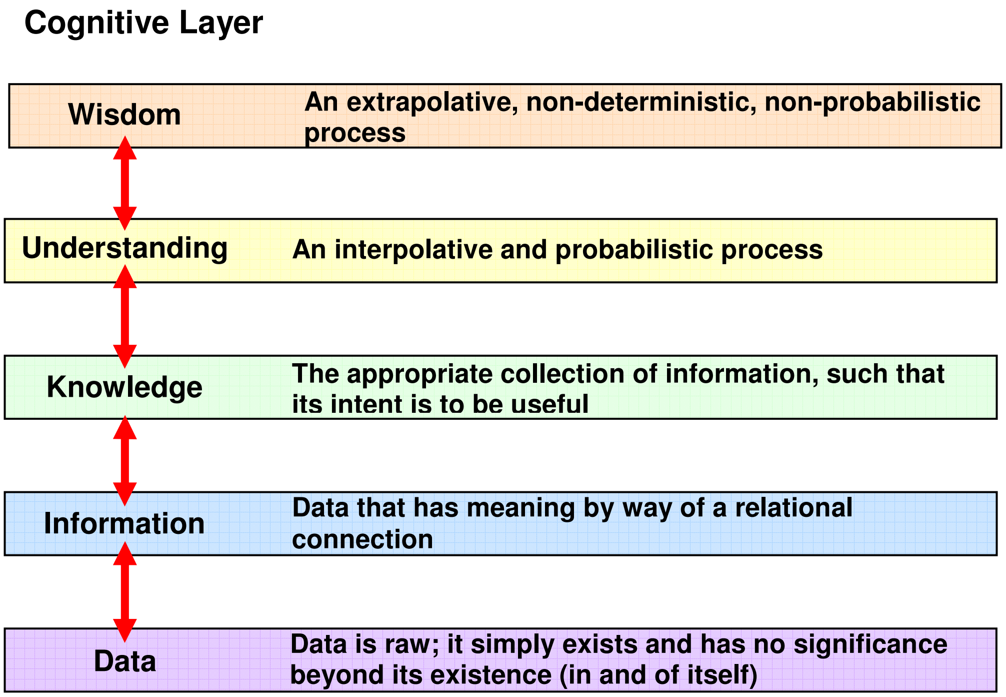

Cognitive Layer
Wisdom
An extrapolative, non-deterministic, non-probabilistic process
Understanding
An interpolative and probabilistic process
Knowledge
The appropriate collection of information, such that its intent is to be useful
Information
Data that has meaning by way of a relational connection
Data
Data is raw; it simply exists and has no significance beyond its existence (in and of itself)

Figure 5. Cognitive Model

1.2.1 Bottom-Up Cognitive Model Example

The simple *Bottom-Up Cognitive Model* example presented in Figure 6 illustrates how bottom-up cognition applies in our lives, usually as part of analytical processes. It is *bottom-up* because the process described starts with *Data* and ends with *Wisdom*. At the *Cognitive Data Layer*, a temperature of 100° means little. Adding that the temperature is in degrees Fahrenheit provides a bit more data; however, it still has little relevance until the temperature is put in the context of a person's temperature and becomes *Information*. Adding that information with other information like the normal temperature for a person is 98.6° Fahrenheit starts to provide us some *Knowledge* of the situation. This knowledge, combined with other knowledge, allows us to understand that the person has the flu. The final step is adding this knowledge with what we already know about the individual allowing a decision that the temperature is not serious and that the solution is to take two aspirin and call the doctor in the morning if symptoms persist. In reality, there is more data than information, more information than knowledge, etc.

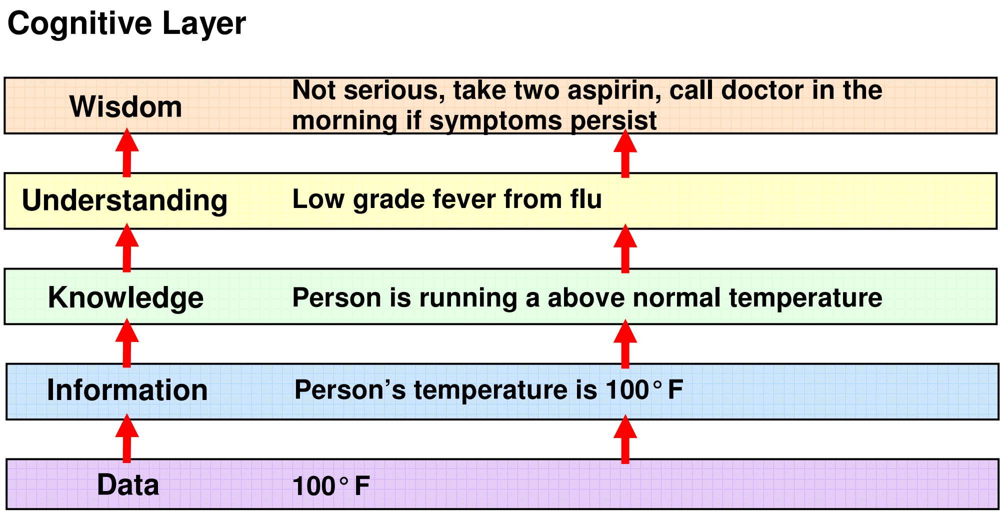

Cognitive Layer
Wisdom
Not serious, take two aspirin, call doctor in the morning if symptoms persist
Understanding
Low grade fever from flu
Knowledge
Person is running a above normal temperature
Information
Person's temperature is 100° F
Data
100° F

Figure 6. Example of Cognitive Model – Data to Wisdom

1.2.2 Top-Down Cognitive Model Example

The simple *Top-Down Cognitive Model* example presented in Figure 7 illustrates how top-down cognition applies in our lives, usually as part of educational or regulatory processes. It is *top-down* because the process described starts with *Wisdom* and ends with *Data*. At the *Cognitive Wisdom Layer*, there needs to be a uniform policy to protect people at risk from influenza. To support this policy (i.e., *Wisdom*), there are many different kinds of things to understand, such as people can be immunized against flu. As a part of the *Understanding*, there is *Knowledge* that vaccines are made from eggs. This leads to the need to disseminate *Information* that people who are allergic to eggs can not use the vaccine and ultimately the collection of *Data* (i.e., evaluation criteria) about egg allergies from people receiving the vaccine.

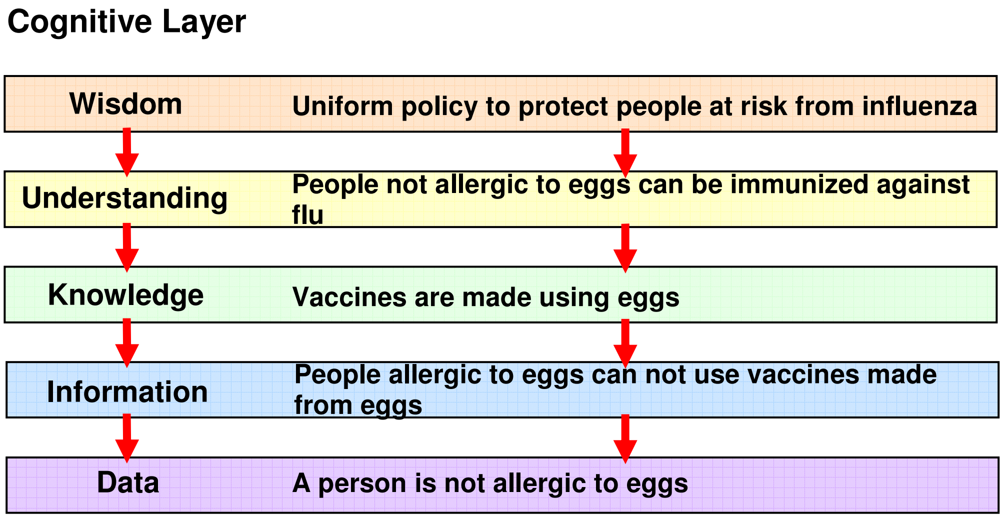
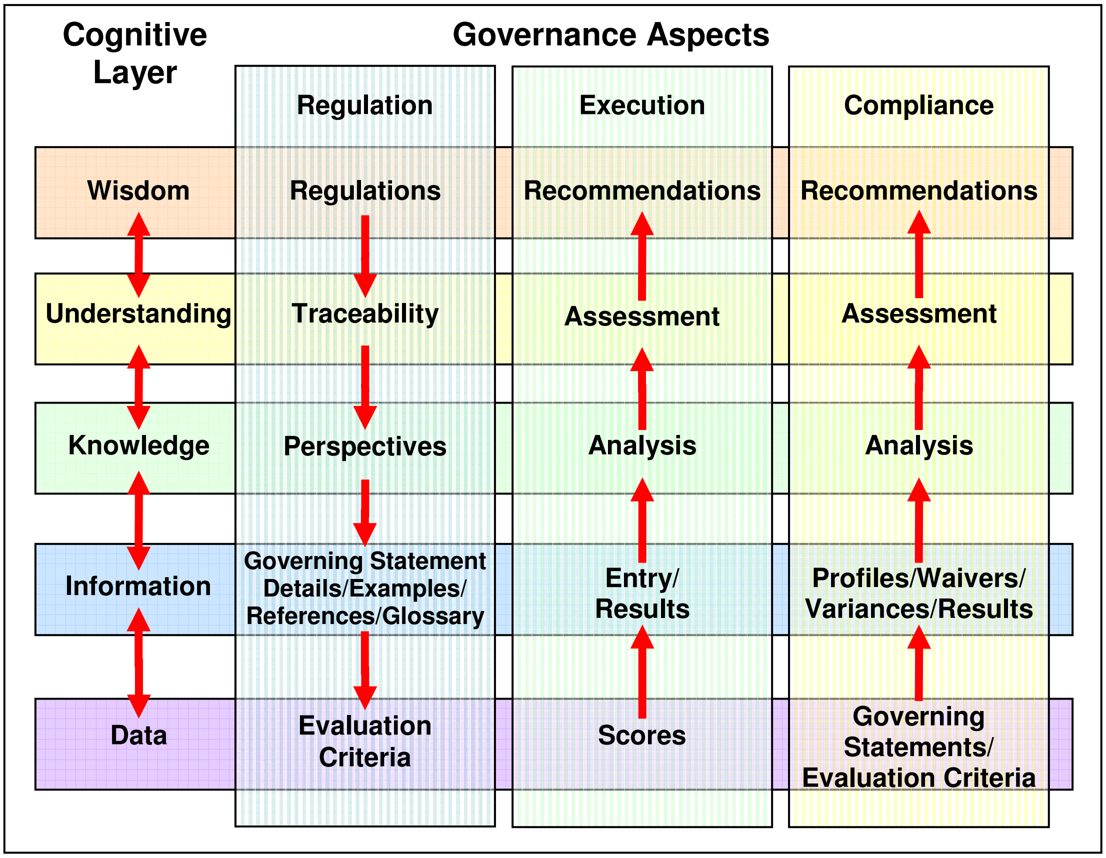

Cognitive Layer
Wisdom
Uniform policy to protect people at risk from influenza
Understanding
People not allergic to eggs can be immunized against flu
Knowledge
Vaccines are made using eggs
Information
People allergic to eggs can not use vaccines made from eggs
Data
A person is not allergic to eggs

Figure 7. Example of Cognitive Model – Wisdom to Data

1.3 Governance Conceptual Model

The *Governance Conceptual Model* brings both the *Fundamental Governance Model* and the *Cognitive Model* together into a single model. Figure 8 is a simplistic view of this combined model. Each of the three *Governance Aspects* is represented as one of the columns superimposed on the Cognitive Layers; the arrows indicate the primarily top-down or bottom-up flow used for that governance aspect. At the intersection of the Cognitive Model Layers and the Governance Model columns is a cell referred to as a *role*. Each role describes metadata and meta-products required to meet that part of governance.



Cognitive Layer	Regulation	Execution	Compliance
Wisdom	Regulations	Recommendations	Recommendations
Understanding	Traceability	Assessment	Assessment
Knowledge	Perspectives	Analysis	Analysis
Information	Governing Statement Details/Examples/ References/Glossary	Entry/ Results	Profiles/Waivers/ Variances/Results
Data	Evaluation Criteria	Scores	Governing Statements/ Evaluation Criteria

Figure 8. Governance Conceptual Model Roles

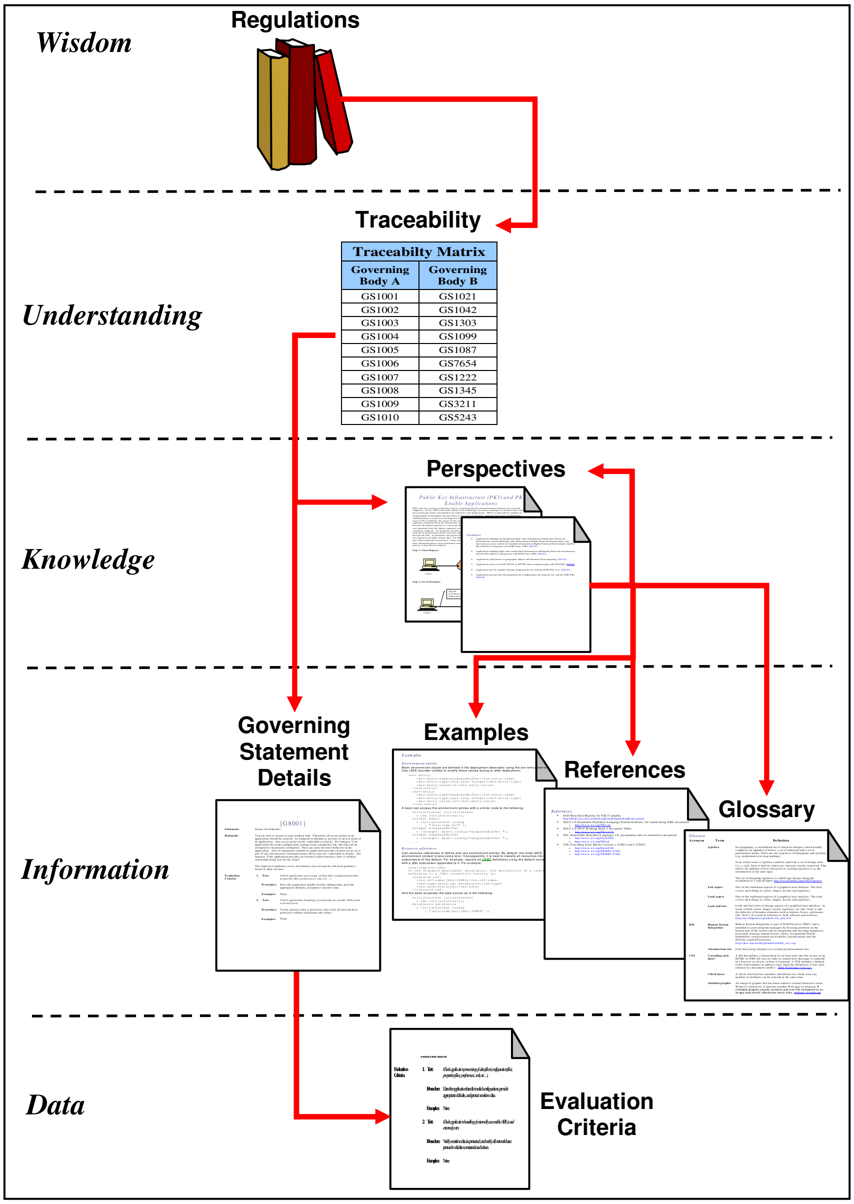
The Governance Conceptual Model fulfils its primary purpose by capturing the governance objects and their relationship to each other. However, some rules apply to use this model successfully (see Appendix A). For example, each of the roles in the Governance Conceptual Model requires a *Data Model* to meet the *Semantic Tagging Rule* (see A.2). Because of the high interdependence of the Cognitive Layers within a Governance Aspect, it is best to create a single Data Model for each of the aspects that covers all the roles.

1.3.1 Regulation Conceptual Data Model

The *Regulation Conceptual Data Model* has at least one object for each of the five layers of the Cognitive Model, mapped as follows.

Table 3. Mapping Cognitive Layers to Regulatory Objects

Wisdom	Regulations
Understanding	Traceability
Knowledge	Perspectives
Information	Governing Statement Details
Data	Evaluation Criteria



Traceabilty Matrix	
Governing Body A	**Governing Body B**
GS1001	GS1021
GS1002	GS1042
GS1003	GS1303
GS1004	GS1099
GS1005	GS1087
GS1006	GS7654
GS1007	GS1222
GS1008	GS1345
GS1009	GS3211
GS1010	GS5243

Figure 9. Regulation Conceptual Data Model

1.3.1.1 Regulations

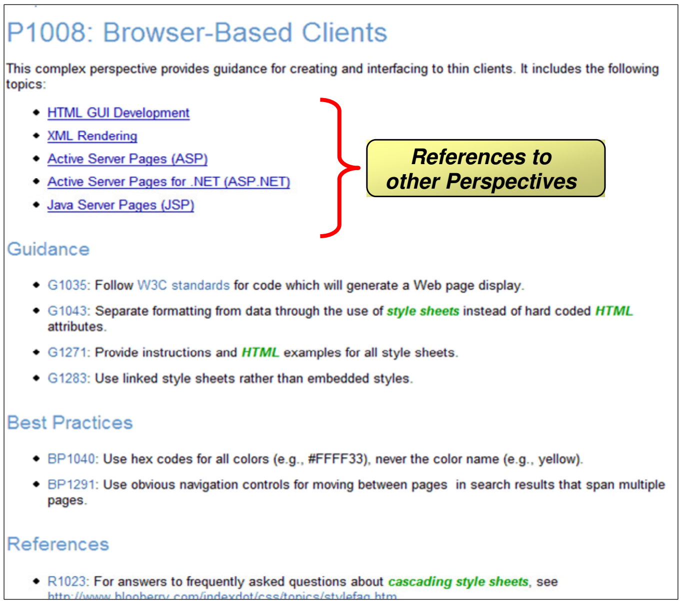
Regulation objects capture two kinds of regulator content: *external* and *internal*. External regulation is that content that is not directly in the control of the governing body; it is generally only structured using formatting and is delivered as a long linear document. Often keywords such as *shall*, *will*, *should* and *may* identify the guidance. Sometimes these regulations are statutes and sometimes they are statutory in nature. There are efforts underway, such as the OMG *Compliance Global Regulatory Information Database (C-GRID)* effort, that are trying to capture the attributes necessary to represent the external regulations adequately. Internal regulations are those created under the purview of the governing body according to the other objects in the Regulatory Conceptual Data Model.

1.3.1.2 Traceability

Often the regulatory *Understanding* layer captures high-level, abstract, hard to measure tenets rather than specific *Governing Statements*. These tenets, however, can map to specific, quantifiable Governing Statements. This is generally a many-to-many relationship with one high-level tenet mapping to multiple Governing Statements and one Governing Statement mapping to multiple tenets. This mapping within the Regulation Aspect of governance can be the basis for the *Summary*, *Analysis* or *Evaluation* roles in the Execution or Compliance aspects of Governance.

1.3.1.3 Perspectives

A *Perspective* is a container for aggregating other Regulation Objects that are related functionally. All Perspectives have a title (the example below includes a unique Perspective ID as part of the title) and a brief description of functionality related to the regulations. The description is not the definitive explanation of the topic but rather a brief, high level description that can point to more complete works. There are two basic types of perspectives: *complex* and *detailed*. *Complex Perspectives* aggregate other Perspective Objects and allow organizing Perspectives into a hierarchy.



P1008: Browser-Based Clients

This complex perspective provides guidance for creating and interfacing to thin clients. It includes the following topics:

- HTML GUI Development
- XML Rendering
- Active Server Pages (ASP)
- Active Server Pages for .NET (ASP.NET)
- Java Server Pages (JSP)

Guidance

- G1035: Follow W3C standards for code which will generate a Web page display.
- G1043: Separate formatting from data through the use of ***style sheets*** instead of hard coded ***HTML*** attributes.
- G1271: Provide instructions and ***HTML*** examples for all style sheets.
- G1283: Use linked style sheets rather than embedded styles.

Best Practices

- BP1040: Use hex codes for all colors (e.g., #FFFF33), never the color name (e.g., yellow).
- BP1291: Use obvious navigation controls for moving between pages in search results that span multiple pages.

References

- R1023: For answers to frequently asked questions about ***cascading style sheets***, see http://www.blooberry.com/indexdot/css/topics/stylefaq.htm

Figure 10. Example of a Complex Perspective.

Detailed Perspectives aggregate other Regulation objects together such as Governing Statements (which in the figures above and below are in the form of Guidance and Best Practices), Examples, References, and Glossary links (shown in the figure as *green italic* text) that are related to the functionality.

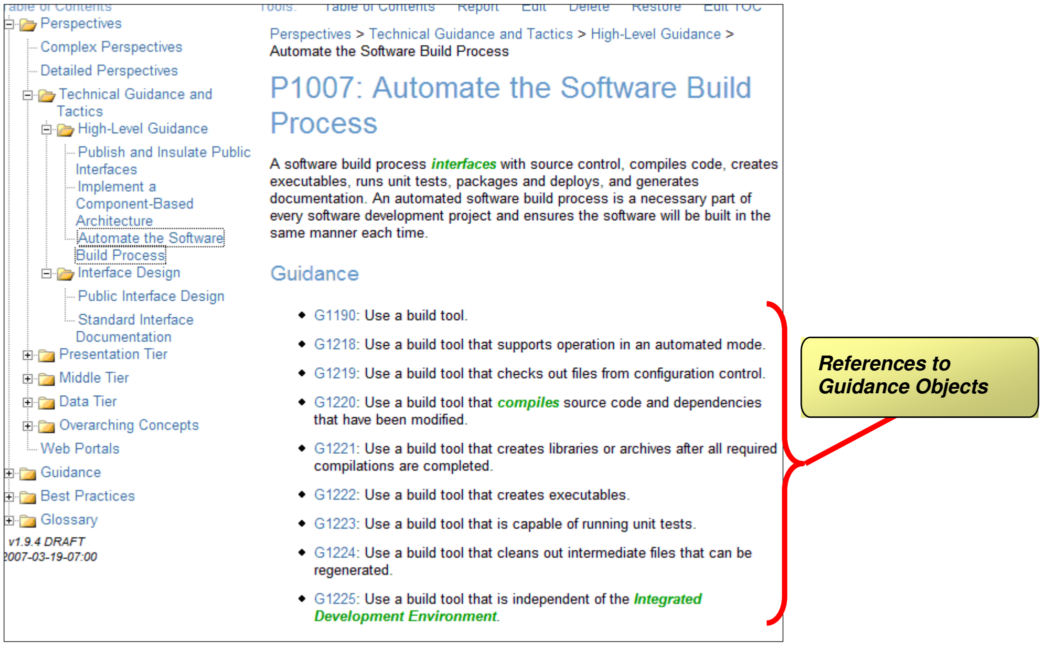


Table of Contents

- Perspectives
  - Complex Perspectives
  - Detailed Perspectives
  - Technical Guidance and Tactics
    - High-Level Guidance
      - Publish and Insulate Public Interfaces
      - Implement a Component-Based Architecture
      - Automate the Software Build Process
    - Interface Design
      - Public Interface Design
      - Standard Interface Documentation
    - Presentation Tier
    - Middle Tier
    - Data Tier
    - Overarching Concepts
  - Web Portals
- Guidance
- Best Practices
- Glossary

v1.9.4 DRAFT
2007-03-19-07:00

Tools: Table of Contents Report Edit Delete Restore Edit TOC

Perspectives > Technical Guidance and Tactics > High-Level Guidance >
Automate the Software Build Process

P1007: Automate the Software Build Process

A software build process ***interfaces*** with source control, compiles code, creates executables, runs unit tests, packages and deploys, and generates documentation. An automated software build process is a necessary part of every software development project and ensures the software will be built in the same manner each time.

Guidance

- G1190: Use a build tool.
- G1218: Use a build tool that supports operation in an automated mode.
- G1219: Use a build tool that checks out files from configuration control.
- G1220: Use a build tool that ***compiles*** source code and dependencies that have been modified.
- G1221: Use a build tool that creates libraries or archives after all required compilations are completed.
- G1222: Use a build tool that creates executables.
- G1223: Use a build tool that is capable of running unit tests.
- G1224: Use a build tool that cleans out intermediate files that can be regenerated.
- G1225: Use a build tool that is independent of the ***Integrated Development Environment***.

Figure 11. Example of Detailed Perspective

1.3.1.4 Governing Statements

Governing Statements capture a specific regulatory concept within a single, active voice sentence and are supported by details which elaborate on the statement. A Governing Statement must be *atomic*, *succinct*, *absolute* and *definitive* in nature.

Atomic A Governing Statement only addresses a single topic. Indicators of non-atomic guidance are use of complex sentences, multiple sentences or conjunctions such as *and*, *or*, etc.

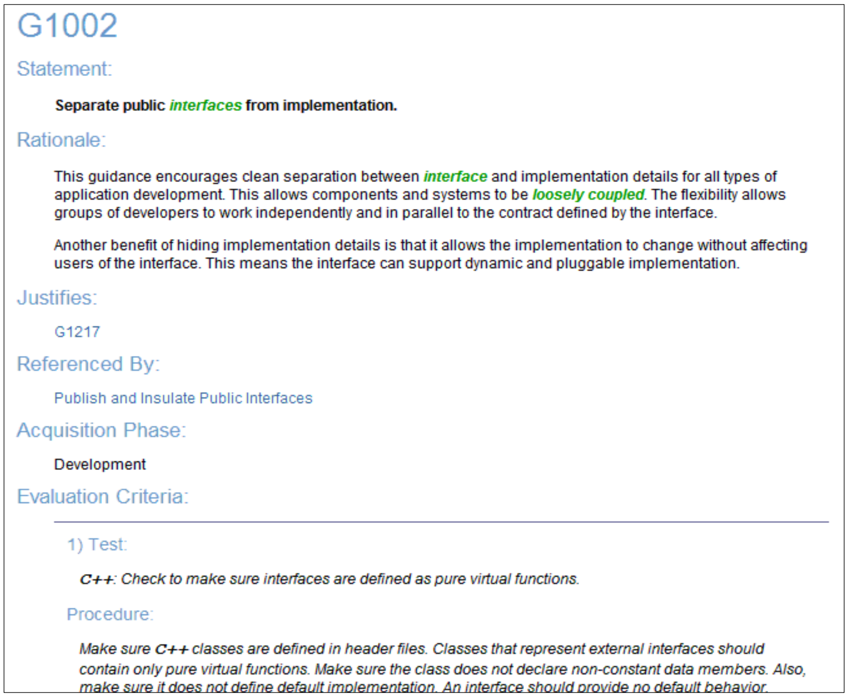
Succinct A Governing Statement is short and to the point. The definition of terms or caveats that explain when a statement is applicable are not acceptable as part of the Governing Statement. Indicators of non-succinct statements are the use of words or expressions such as: *consider*, *when possible*, *if*, etc.

Absolute A Governing Statement is subject to evaluation with one or more non-subjective questions. Indicators of non-absolute statements are those which are subject to the interpretation of the evaluator. For example, “All menus must be user friendly." Software developers do not produce menus that they feel are hostile.

Definitive A Governing Statement is precisely worded and explicit in nature. The words, terms and expressions need known definitions, not subject to conflicting interpretation. Indicators of non-definitive words are that they are not intuitively obvious to an outside reader. Some words that are examples of non-explicit words are *object*, *service* and *function*.

Governing Statements contain specific instructions which can be validated through observation, measurement or testing (see A.3). Define any words that are not standard English usage in the Glossary. Governing Statements use the active voice and do not rely on the use of key words such as *will*, *should*, *shall* or *may* to convey a requirement. If this is necessary, then preceding the statements with the appropriate key word is sufficient as in the following example:

> *The contractor shall conform to Governing Statements G1002 and G1324.*



G1002

Statement:

Separate public *interfaces* from implementation.

Rationale:

This guidance encourages clean separation between *interface* and implementation details for all types of application development. This allows components and systems to be *loosely coupled*. The flexibility allows groups of developers to work independently and in parallel to the contract defined by the interface.

Another benefit of hiding implementation details is that it allows the implementation to change without affecting users of the interface. This means the interface can support dynamic and pluggable implementation.

Justifies:

G1217

Referenced By:

Publish and Insulate Public Interfaces

Acquisition Phase:

Development

Evaluation Criteria:

1) Test:

C++: Check to make sure interfaces are defined as pure virtual functions.

Procedure:

Make sure C++ classes are defined in header files. Classes that represent external interfaces should contain only pure virtual functions. Make sure the class does not declare non-constant data members. Also, make sure it does not define default implementation. An interface should provide no default behavior.

Figure 12. Example of Governing Statement Details

Regulations translate into a collection of inter-related Governing Statements that cover the spectrum from statutes and high-level goals and objectives down to detailed prescriptive instructions of how to operate.

1.3.1.5 Rationale

The *Rationale* is a brief explanation of why the Governing Statement was formulated. This is not a major treatise on the subject covered by the Governing Statement, but rather a high-level summary. Specific details, if required, are referenced in other documents, white papers, mandates or standards.

There are two reasons why the Rationale is so important: *Waivers* and *Profiles*. An outside reviewer analyzes the justification for a Waiver against the Rationale. A Community of Interest uses the Rationale when considering which Governing Statements to include in a particular Profile.

1.3.1.6 Cross References

There are several ways to cross reference the Governing Statements. To support the hierarchical relationships, references point to *parents* (i.e., where it is derived from) and *children* (i.e., what it justifies).

The *Derived From* references include links to the parents of a Governing Statement. There can be any number of such links; the only limitation is that ultimately a child of a statement cannot also be a parent of the same statement.

The *Justifies* references include links to the children of a Governing Statement. There can be any number of such links; the only limitation is that ultimately a child cannot also be a parent of the same Governing Statement.

1.3.1.7 Life Cycle Phase

Most Governing Statements are associated with one or more keywords that help categorize the statements for future retrieval. In the Governing Statement Detail example in Figure 12 above, the *Development* tag indicates the DoD Acquisition Phase is associated with the Governing Statement.

1.3.1.8 Evaluation Criteria

A Governing Statement is subject to evaluation through observation, measurement or other testing. The *Evaluation Criteria* prescribe the methods to use to accomplish this. The published Evaluation Criteria provide a way of determining that the Governing Statement was actually met. There may be other ways of determining compliance; document these and submitted recommended changes or additions to the Governing Body or other responsible organization. Evaluation Criteria include three parts.

Test	A direct question about the Governing Statement
Procedure	A process or procedure to follow to determine the answer to the Test question
Examples	Optional pieces of code, text or graphics that illustrate the Test or Procedure

1.3.1.8.1 Tests

A Governing Statement *Test* is usually a question formulated from the statement. The answer to the question can help determine if the intent of the statement has been met. There are several different categories of questions. *Yes/No* or *True/False* questions indicate completion. *Level of Compliance* questions indicate that there can be variation in the rigor of compliance as in the following levels:

No Compliance No attempt to meet the Governing Statement intent.

Low Compliance Very few of the possible places of where the intent of the Governing Statement could be implemented are implemented.

Some Compliance Many of the possible places of where the intent of the Governing Statement could be implemented are implemented.

Complete Compliance The implementation meets the full intent of the Governing Statement.

The absolute levels (i.e., *No* or *Complete*) are easy to determine. The differentiation between *Low* and *Some* is more subjective. A good tenet would be the 50% mark.

1.3.1.8.2 Procedures

A Governing Statement *Procedure* helps increase the repeatability of the evaluation process. It defines a specific way to derive an answer to the Test question from observing the program or project deliverables. A goal is that once formulated, these procedures can be automated. This will eliminate the subjective nature of the evaluation as well as increase the speed and reduce the cost of performing evaluations.

1.3.1.8.3 Examples

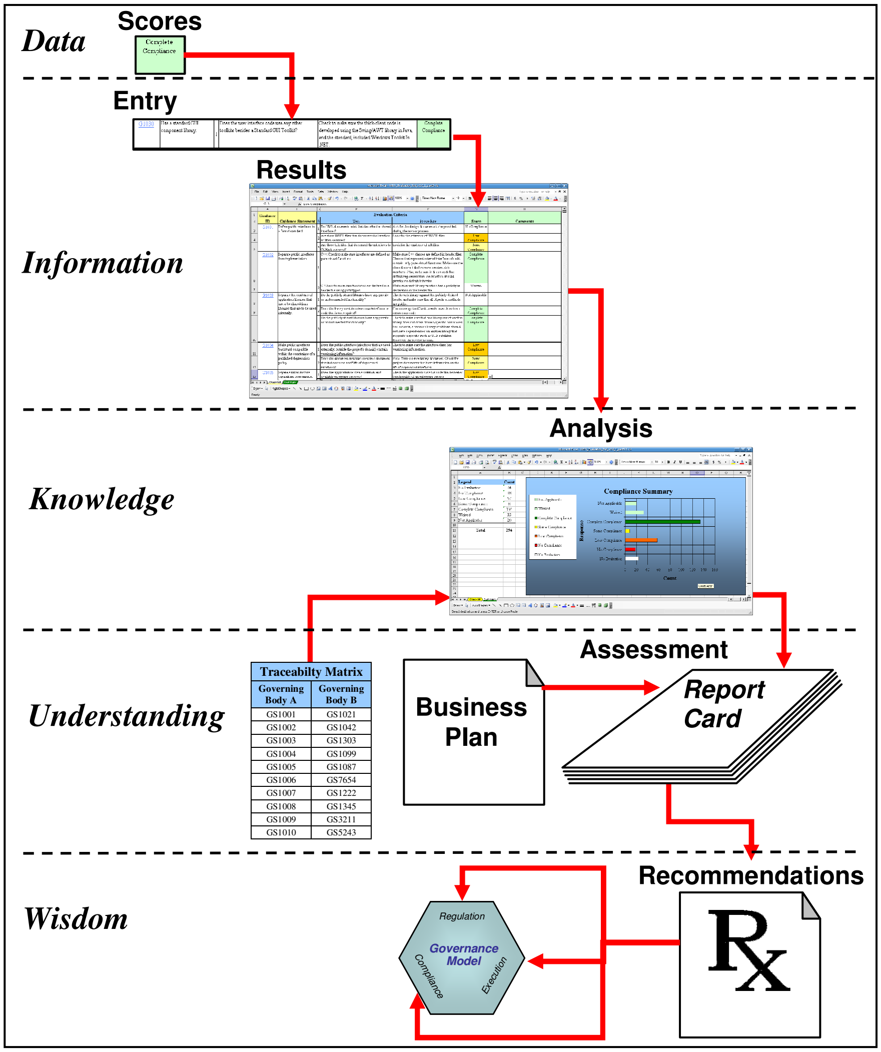
A Governing Statement *Example* illustrates what will (or will not) pass when using the Evaluation Criteria Procedure as an aid for the evaluator.

1.3.2 Execution Conceptual Data Model

The *Execution Conceptual Data Model* has at least one object for each of the five layers of the Cognitive Model, mapped as follows.

Table 4. Mapping Cognitive Layers to Execution Objects

Data	Scores
Information	Entry/Results
Knowledge	Analysis
Understanding	Assessment
Wisdom	Recommendations



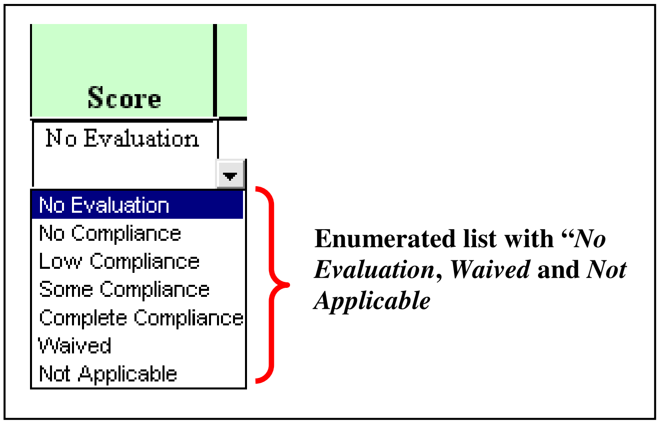
Traceabilty Matrix	
Governing Body A	Governing Body B
GS1001	GS1021
GS1002	GS1042
GS1003	GS1303
GS1004	GS1099
GS1005	GS1087
GS1006	GS7654
GS1007	GS1222
GS1008	GS1345
GS1009	GS3211
GS1010	GS5243

Figure 13. Execution Conceptual Data Model

1.3.2.1 Scores

Scores are raw numeric data values assigned to reflect compliance with a particular Governing Statement's Evaluation Criteria. Scores alone do little for providing information, knowledge, understanding or wisdom of compliance with a Governing Statement. It is only when the scores are put in context of the Governing Statements that the scores are useful. Discrete numeric values, usually represented by an enumerated type, represent scores; for example, *No Compliance*, *Low Compliance*, *Some Compliance*, and *Complete Compliance*. Sometimes the enumeration can be reduced to Boolean values such as *Yes/No* or *True/False*. In addition to a prescribed enumerated list of answers, scores allow for *Not Applicable*, *Waived*, and *No Evaluation* values. Sometimes the null value is substituted for *No Evaluation*.

Scores that are purely numeric in nature such as temperature or percentile need to have defined ranges and also support the *Not Applicable*, *Waived* and *No Evaluation* values.



Score

No Evaluation

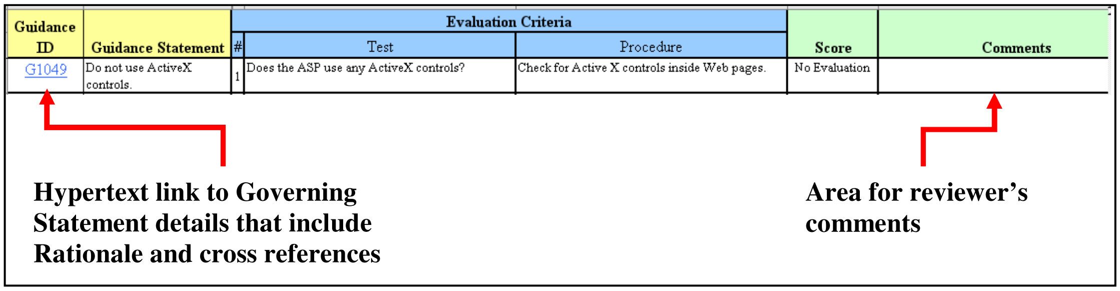
- No Evaluation
- No Compliance
- Low Compliance
- Some Compliance
- Complete Compliance
- Waived
- Not Applicable

Enumerated list with "*No Evaluation*, *Waived* and *Not Applicable*

Figure 14. Example of Scoring Enumeration

1.3.2.2 Entries

Entries are complex information structures that not only contain the score but provide the contextual information needed to associate the score with the regulating Governing Statement and the appropriate Evaluation Criteria. The following example illustrates how a single entry provides the Evaluation Criteria and the Governing Statement text as well as a field for comments and a hypertext link back to the original Governing Statement where the Rationale and the cross reference information are available.



Guidance ID	Guidance Statement	#	Evaluation Criteria		Score	Comments
			Test	Procedure		
G1049	Do not use ActiveX controls.	1	Does the ASP use any ActiveX controls?	Check for Active X controls inside Web pages.	No Evaluation	

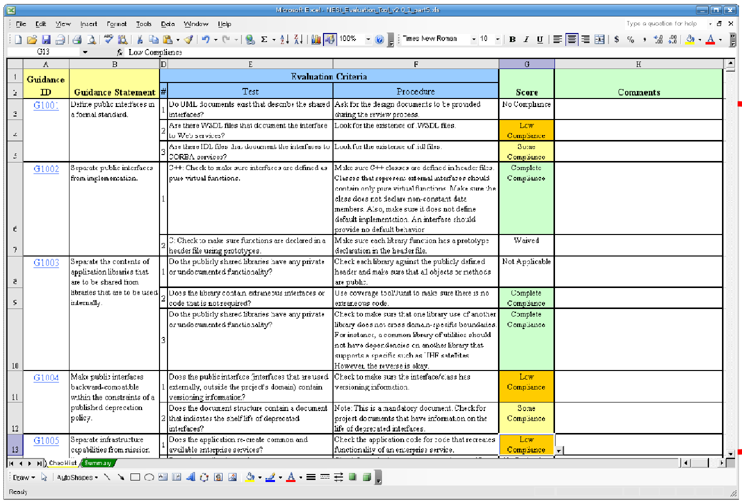
Hypertext link to Governing Statement details that include Rationale and cross references

Area for reviewer's comments

Figure 15. Example of Evaluation Entry

1.3.2.3 Results

Results are collections of entries. There may not be an entry for every Governing Statement or for all the Evaluation Criteria for any particular execution. Governing Statements and Evaluation Criteria are included in the results based on the profile of a particular execution. Governing Statements are independent of the profile inclusion. Therefore, a Governing Statement uses active voice without disclaimers about when it needs to be included. For example, Governing Statements can be C++ or .NET specific and not have to be prefaced with "If using C++" or "If using .NET." It is the responsibility of the profile to eliminate the unnecessary Governing Statements and corresponding Evaluation Criteria. The inclusion or exclusion Governing Statements or Evaluation Criteria is a business decision.


Microsoft Excel - NESI_Evaluation_Tool_v2.0_1_part5.xls
File Edit View Insert Format Tools Data Window Help
Type a question for help
100%
Times New Roman
10
G13
Low Compliance
Checklist
Summary
Draw
AutoShapes
Ready

Guidance ID	Guidance Statement	#	Test	Procedure	Score	Comments
G1001	Define public interfaces in a formal standard.	1	Do UML documents exist that describe the shared interfaces?	Ask for the design documents to be provided during the review process.	No Compliance	
		2	Are there WSDL files that document the interface to Web services?	Look for the existence of .WSDL files.	Low Compliance	
		3	Are there IDL files that document the interfaces to CORBA services?	Look for the existence of .idl files.	Some Compliance	
G1002	Separate public interfaces from implementation.	1	C++: Check to make sure interfaces are defined as pure virtual functions.	Make sure C++ classes are defined in header files. Classes that represent external interfaces should contain only pure virtual functions. Make sure the class does not declare non-constant data members. Also, make sure it does not define default implementation. An interface should provide no default behavior	Complete Compliance	
		2	C: Check to make sure functions are declared in a header file using prototypes.	Make sure each library function has a prototype declaration in the header file.	Waived	
G1003	Separate the contents of application libraries that are to be shared from libraries that are to be used internally.	1	Do the publicly shared libraries have any private or undocumented functionality?	Check each library against the publicly defined header and make sure that all objects or methods are public.	Not Applicable	
		2	Does the library contain extraneous interfaces or code that is not required?	Use coverage tool/Junit to make sure there is no extraneous code.	Complete Compliance	
		3	Do the publicly shared libraries have any private or undocumented functionality?	Check to make sure that one library use of another library does not cross domain-specific boundaries. For instance, a common library of utilities should not have dependencies on another library that supports a specific such as UHF satellites. However, the reverse is okay.	Complete Compliance	
G1004	Make public interfaces backward-compatible within the constraints of a published deprecation policy.	1	Does the public interface (interfaces that are used externally, outside the project's domain) contain versioning information?	Check to make sure the interface/class has versioning information.	Low Compliance	
		2	Does the document structure contain a document that indicates the shelf life of deprecated interfaces?	Note: This is a mandatory document. Check for project documents that have information on the life of deprecated interfaces.	Some Compliance	
G1005	Separate infrastructure capabilities from mission.	1	Does the application re-create common and available enterprise services?	Check the application code for code that recreates functionality of an enterprise service.	Low Compliance	

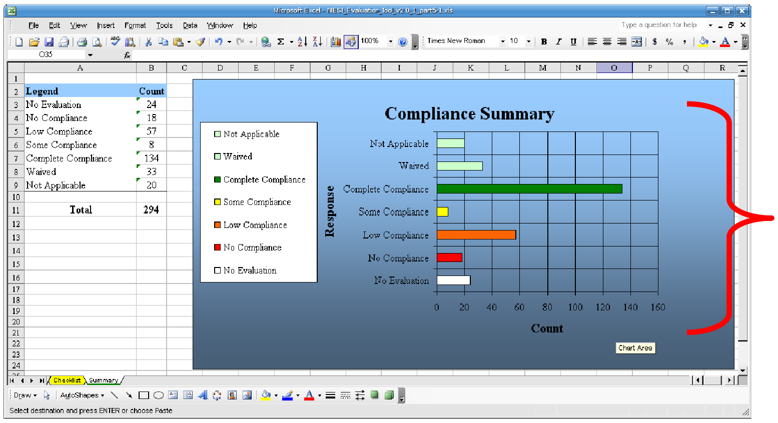
Results are a collection of Entries

Figure 16. Example of Results

1.3.2.4 Analysis

The goal of *Analysis* is to impart knowledge about an Execution's adherence to applicable (based on profile) Governing Statements. Much of the analysis is statistical in nature and includes sums, means, variance, and standard deviations of the scores in the results. The statistics are built around formal and informal taxonomic classifications of Governing Statements and Evaluation Criteria. Communities of Interest (COI) develop and publish formal taxonomies or ontologies or the taxonomies or ontologies are present inherently in the presentation of the regulations within Perspectives. One example of a formal taxonomy of Governing Statements is the set of statements in a particular Perspective or its nodal pathway (see A.3).

Examples of informal taxonomies are folksonomies or tags that analysts develop to measure effectiveness of the Governing Statements on the execution.



Legend	Count
No Evaluation	24
No Compliance	18
Low Compliance	57
Some Compliance	8
Complete Compliance	134
Waived	33
Not Applicable	20
Total	294

Analysis includes sums, means, variance and standard deviations of scores around traceability or taxonomic classifications of Governing Statements

Figure 17. Example of Analysis

1.3.2.5 Assessments

Assessments use analytic knowledge to develop an understanding of the execution's adherence to the Governing Statements. The assessment includes a detailed discussion of the strengths and weaknesses of the execution and any potential cause and effects for the compliance and variances especially as they relate to formal taxonomies or ontologies. The assessments are not in terms of *pass* or *fail* but in terms of figures of merit (FOM) which represent a measure of the effectiveness of the execution in terms of the formal taxonomies or ontologies. The assessment documents reasons for any particular FOM value, such as the speed of refresh or business decision tradeoffs.

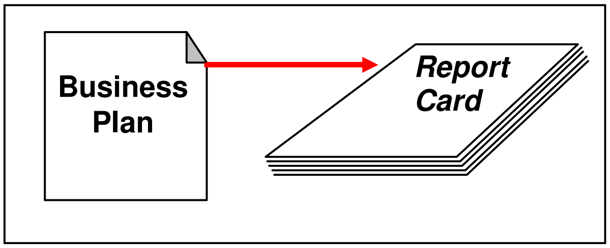


Figure 18. Example of Assessment

1.3.2.6 Recommendations

Recommendations are based on the assessment of the execution. A recommendation can apply to any of the aspects of the Governance Model: Regulation, Execution or Compliance.

Regulation Recommendations occur when the regulations are poor at providing effective direction. The Regulation aspect of Governance does not necessarily adopt Regulation Recommendations but may adjudicate the recommendations in future regulation efforts.

Execution Recommendations occur when there are deficiencies in the overall execution of the regulations as determined by Compliance. The Execution aspect of Governance does not necessarily act upon Execution Recommendations but may adjudicate the recommendations in future iterations of the execution effort. Even given infinite amounts of time and effort, the reality of the Business Plan may preclude acting on all Execution Recommendations.

Compliance Recommendations occur when the FOMs are poor or misleading for determining overall compliance with the regulations. The Compliance aspect of Governance does not necessarily adopt Compliance Recommendations but may adjudicate the recommendations in future compliance efforts.

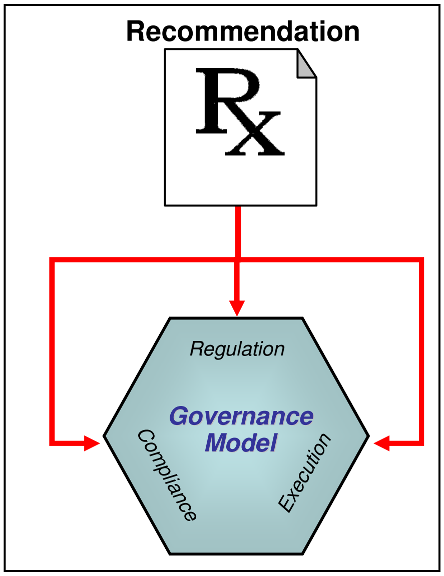


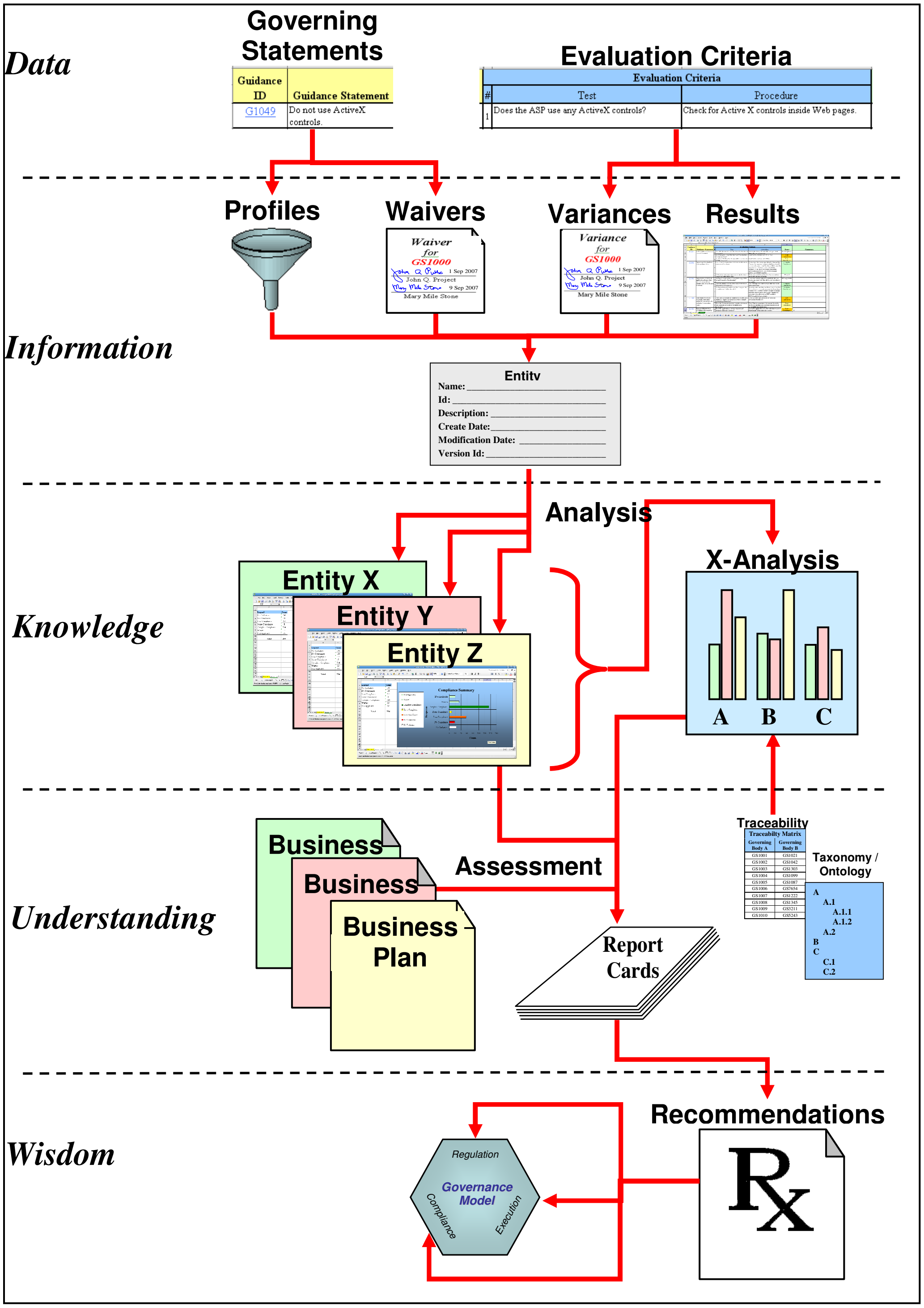
Figure 19. Example of Recommendation

1.3.3 Compliance Conceptual Data Model

The *Compliance Conceptual Data Model* has at least one object for each of the five layers of the Cognitive Model, mapped as follows.

Table 5. Mapping Cognitive Layers to Execution Objects

Data	Governing Statements, Evaluation Criteria
Information	Profile, Waivers, Variances and Results
Knowledge	Analysis
Understanding	Assessment
Wisdom	Recommendations



Guidance ID	Guidance Statement
G1049	Do not use ActiveX controls.

Evaluation Criteria		
#	Test	Procedure
1	Does the ASP use any ActiveX controls?	Check for Active X controls inside Web pages.

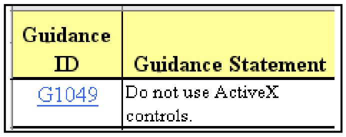
Traceabilty Matrix	
Governing Body A	Governing Body B
GS1001	GS1021
GS1002	GS1042
GS1003	GS1303
GS1004	GS1099
GS1005	GS1087
GS1006	GS7654
GS1007	GS1222
GS1008	GS1345
GS1009	GS3211
GS1010	GS5243

Figure 20. Compliance Conceptual Data Model

1.3.3.1 Governing Statements

In the Compliance aspect of the Governance Model, *Governing Statements* are reference data that point back to the original Governing Statements maintained by the Regulation aspect of the Governance Model. Since the Regulation and the Compliance aspects of the

Governance Model evolve independently, the Governing Statement reference includes the versioning information required to maintain the integrity of the Compliance Model.

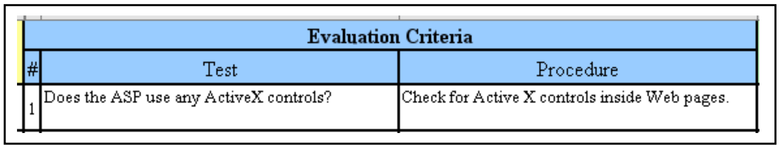


Guidance ID	Guidance Statement
G1049	Do not use ActiveX controls.

Figure 21. Governing Statement Example

1.3.3.2 Evaluation Criteria

In the Compliance aspect of the Governance Model, *Evaluation Criteria* are reference data that point back to the original Evaluation Criteria maintained by the Regulation aspect of the Governance Model. The Regulation aspect of the model is responsible for maintaining the relationship between the Governing Statement and the Evaluation Criteria. Since the Regulation and the Compliance aspects of the Governance Model evolve independently, the Evaluation Criteria Reference includes the versioning information required to maintain the integrity of the Compliance Model.



	Evaluation Criteria	
#	Test	Procedure
1	Does the ASP use any ActiveX controls?	Check for Active X controls inside Web pages.

Figure 22. Evaluation Criteria Reference Example

1.3.3.3 Profiles

A *Profile* is information about which Governing Statements are germane to an Entity and which ones are waived. By default, all Governing Statements are required unless specifically waived using a Waiver. A Profile is classified as information because it is a collection of Governing Statement references that only make sense in the context of an Entity.

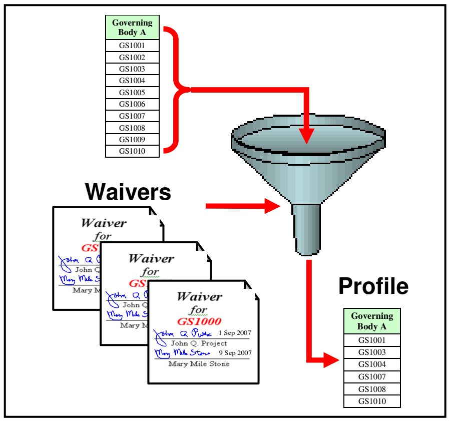


Figure 23. Profile Example

1.3.3.4 Waivers

A *Waiver* is information about a Governing Statement that is not germane to an Entity. It is comprised of a Governing Statement reference and list of the Authorizing Agents, defined by the Agents' names, the date of the approval and signatures.

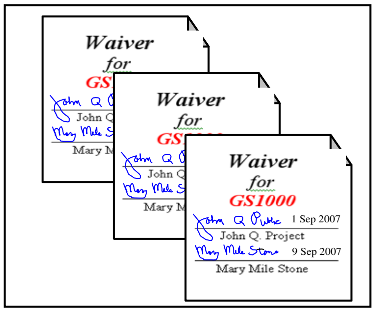
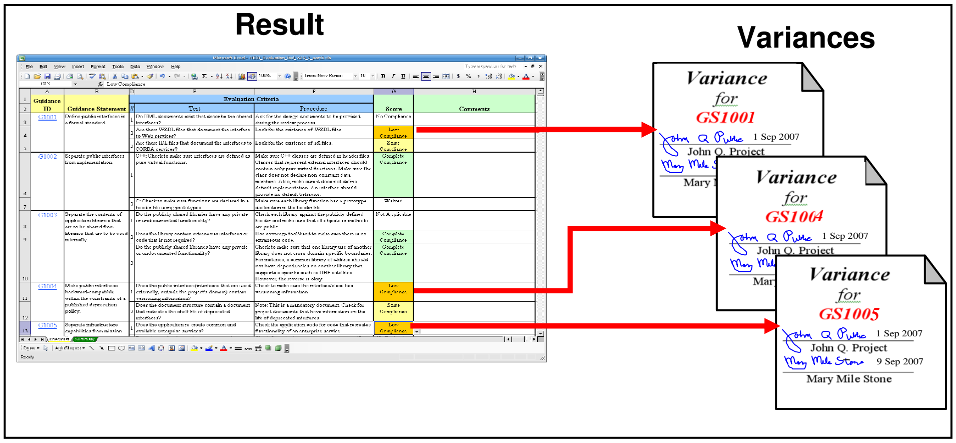


Figure 24. Waiver Examples

1.3.3.5 Results

A *Result* is information about which Evaluation Criteria score references are subject to literal interpretation and which ones have a justification for varying from the standard Evaluation Criteria scores. By default, all Evaluation Criteria are subject to literal interpretation unless they are allowed to vary using a Variance. A Variance is classified as information because it is a collection of Evaluation Criteria score references that only make sense in the context of an Entity.

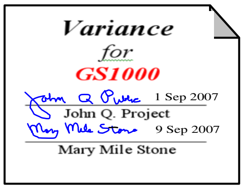


Guidance ID	Guidance Statement	#	Evaluation Criteria Test	 Procedure	Score	Comments
G1001	Define public interfaces in a formal standard.	1	Do UML documents exist that describe the shared interfaces?	Ask for the design documents to be provided during the review process.	No Compliance	
		2	Are there WSDL files that document the interface to Web services?	Look for the existence of .WSDL files.	Low Compliance	
		3	Are there IDL files that document the interfaces to CORBA services?	Look for the existence of idl files.	Some Compliance	
G1002	Separate public interfaces from implementation.	1	C++: Check to make sure interfaces are defined as pure virtual functions.	Make sure C++ classes are defined in header files. Classes that represent external interfaces should contain only pure virtual functions. Make sure the class does not declare non constant data members. Also, make sure it does not define default implementation. An interface should provide no default behavior.	Complete Compliance	
		2	C: Check to make sure functions are declared in a header file using prototypes.	Make sure each library function has a prototype declaration in the header file.	Waived	
G1003	Separate the contents of application libraries that are to be shared from libraries that are to be used internally.	1	Do the publicly shared libraries have any private or undocumented functionality?	Check each library against the publicly defined header and make sure that all objects or methods are public.	Not Applicable	
		2	Does the library contain extraneous interfaces or code that is not required?	Use coverage tool/Junit to make sure there is no extraneous code.	Complete Compliance	
		3	Do the publicly shared libraries have any private or undocumented functionality?	Check to make sure that one library use of another library does not cross domain specific boundaries. For instance, a common library of utilities should not have dependencies on another library that supports a specific such as UHF satellites. However, the reverse is okay.	Complete Compliance	
G1004	Make public interfaces backward-compatible within the constraints of a published deprecation policy.	1	Does the public interface (interfaces that are used externally, outside the project's domain) contain versioning information?	Check to make sure the interface/class has versioning information.	Low Compliance	
		2	Does the document structure contain a document that indicates the shelf life of deprecated interfaces?	Note: This is a mandatory document. Check for project documents that have information on the life of deprecated interfaces.	Some Compliance	
G1005	Separate infrastructure capabilities from mission	1	Does the application re create common and available enterprise services?	Check the application code for code that recreates functionality of an enterprise service.	Low Compliance	

Figure 25. Example of Results

1.3.3.6 Variances

A *Variance* is information about Evaluation Criteria scores that, for some reason, cannot use the Evaluation Criteria Tests or Procedures to determine appropriate scores. It is comprised of a Governing Statement score reference, a list of the Authorizing Agents, and a suggested score that meets the data restrictions on the Evaluation Criteria score. If it cannot meet the data restrictions for the Evaluation Criteria score, calculate the score as a null value. An Authorizing Agent is defined by the Agent's name, the date of the approval and a signature.



Variance
for
GS1000

John Q Public 1 Sep 2007
John Q. Project

Mary Mile Stone 9 Sep 2007
Mary Mile Stone

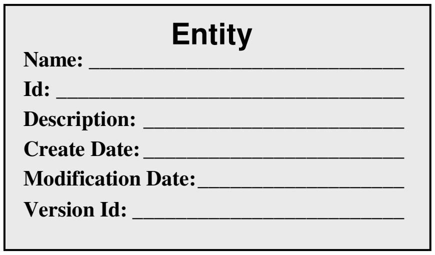
Figure 26. Variance Example

1.3.3.7 Entities

An *Entity* is responsible for the Execution of the Regulations. Entities are often dependent on other Entities executing the same regulations. However, Entities are not authorized to enforce compliance of regulations on other Entities nor are they privy to the execution details of the other entities. For example, a person driving a vehicle on the highway is a Motorist Entity required to follow all the driving regulations. That motorist does not enforce driving regulations on other motorists and has no real knowledge of other motorists on the highway other than to assume they are following the same regulations. Police are responsible for enforcing compliance of the regulations on all drivers and the government is responsible for creating and maintaining the regulations.

A more abstract example of an Entity is a service operating within a Service-Oriented Architecture (SOA). The SOA architects expect the service entities to follow all the rules they set while the SOA environment is responsible for the enforcement of the SOA rules.

An Entity is comprised of a Name, Identifier, Description, Creation Date, Modification Date, and a unique Version Id. Optionally, the Entity can include a set of external references that further define the entity. For example, the motorist may have a driver license number and the service in the SOA may have a URL that points to the Web Service Definition Language (WSDL) descriptor and another URL that points to a human readable Web page that describes the benefits of the particular service.



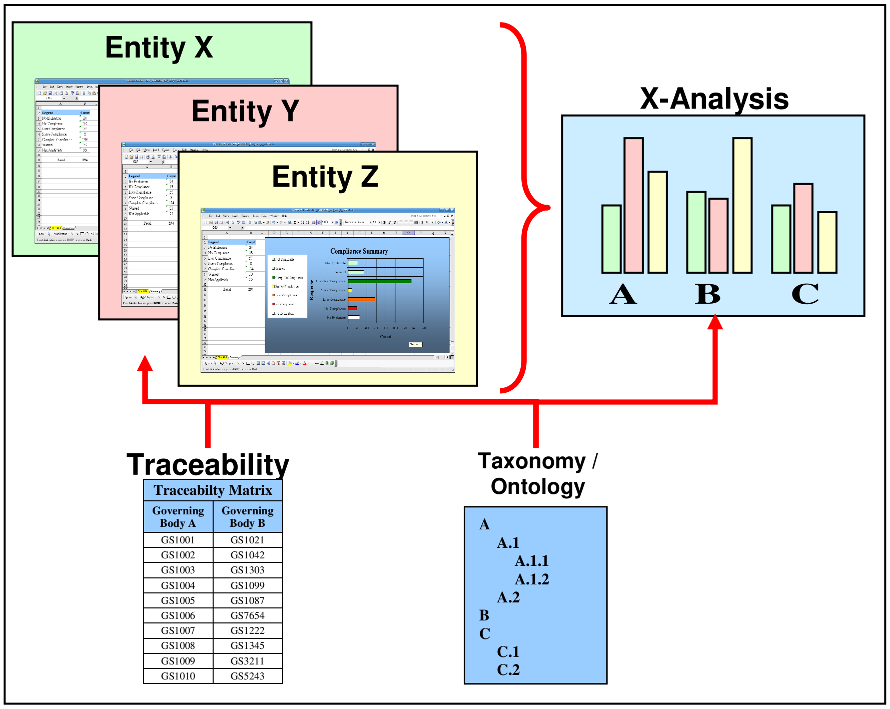
Entity

Name: ______________________________
Id: _________________________________
Description: ________________________
Create Date: ________________________
Modification Date: ___________________
Version Id: __________________________

Figure 27. Example of Entity

1.3.3.8 Entity Analysis

The goal of *Entity Analysis* is to impart knowledge about the Execution's adherence to the Governing Statements relative to other Governance efforts (i.e., traceability), taxonomies, and other Entities. Much of the analysis is statistical in nature and includes sums, means, variance and standard deviations of the scores (see 1.3.2.4 above). Examples of informal taxonomies are folksonomies or tags that analysts develop to measure effectiveness of the Governing Statements on the Execution.



Traceabilty Matrix

Governing Body A	Governing Body B
GS1001	GS1021
GS1002	GS1042
GS1003	GS1303
GS1004	GS1099
GS1005	GS1087
GS1006	GS7654
GS1007	GS1222
GS1008	GS1345
GS1009	GS3211
GS1010	GS5243

Figure 28. Entity Analysis

1.3.3.9 Entity Assessments

Entity Assessments use analytic knowledge to develop an understanding the adherence of individual Entity Executions to the Governing Statements. The assessment includes a detailed discussion of the strengths and weaknesses of each Entity's execution and any potential cause and effects for the compliance and variances especially as they relate to other Entities. The assessments are prescriptive in nature and intended to coordinate Business Plans of all the Entities assessed.

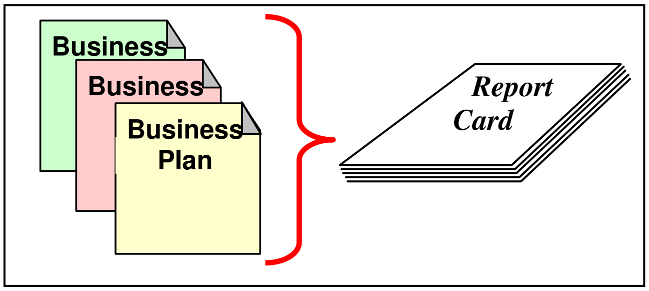


Figure 29. Assessment Example

1.3.3.10 Entity Recommendations

Entity Recommendations are based on Assessments of the Entity's Execution. The recommendations can address any of the aspects of the Governance Model: Regulation, Execution or Compliance.

Entity Regulation Recommendations occur when the regulations are poor at providing effective direction. The Regulation aspect of Governance does not necessarily adopt Entity Regulation Recommendations but may adjudicate the recommendations in future regulation efforts.

Entity Execution Recommendations occur when there are deficiencies in the overall execution of the regulations as determined by Compliance. The focus is on how the adequacies and deficiencies of an Entity relate to the impact on other Entities in the Entity Assessments.

Entity Compliance Recommendations occur when the FOMs are poor or misleading for determining overall compliance with the regulations. The Compliance aspect of Governance does not necessarily adopt Compliance Recommendations but may adjudicate the recommendations in future compliance efforts.

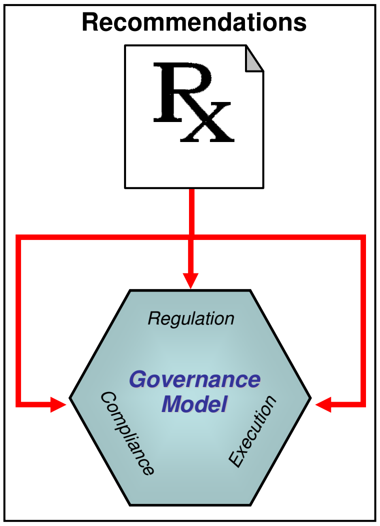


Figure 30. Recommendation Example

1.4 *Executing the Governance Model Regulations*

The Governance Model allows Governing Bodies to publish and maintain regulations and yet share them with the Entities that need to execute those regulations with confidence in a distributed, non-hierarchical, dynamic environment using *Governance Metadata*.

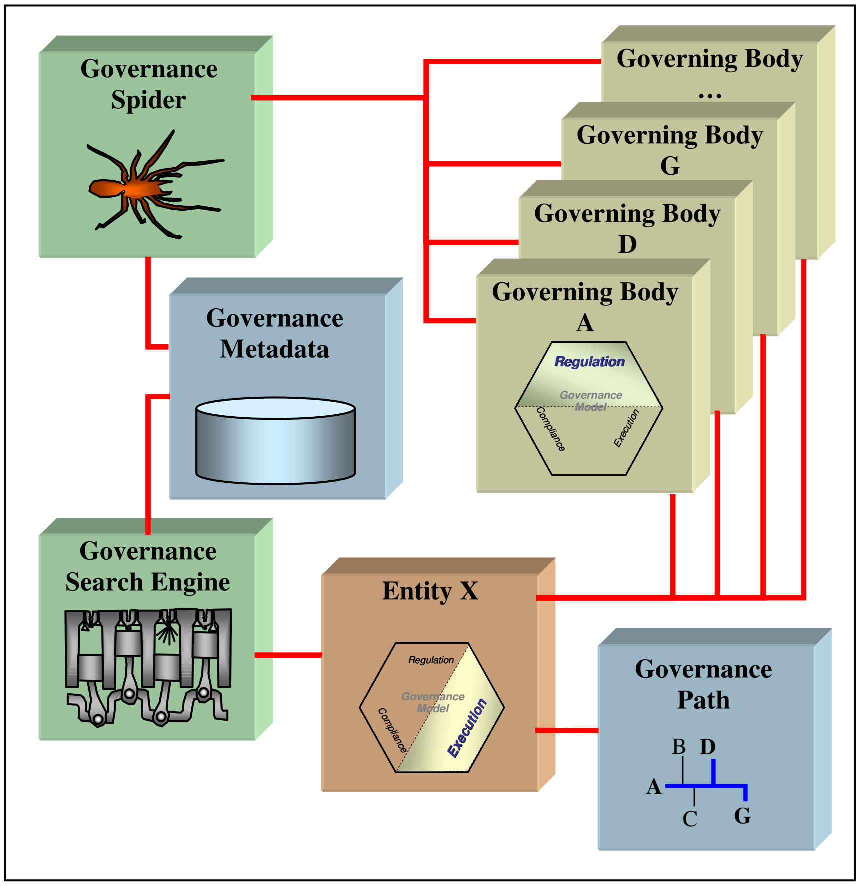


Figure 31. Implementing the Governance Model

1.4.1 Governing Bodies

Governing Bodies are responsible for creating regulations and Entities are responsible for executing those regulations. Often an Entity is subject to any number of Governing Bodies and their regulations. The Governing Bodies that exercise influence over an Entity are not necessarily hierarchical in nature. For example, a business operating within a city must comply with the regulations of the Federal, State and Local governments but in addition, they might be subject to another State’s laws and various standards organizations such as ISO and ANSI.

Governing Bodies expose their regulations as semantically tagged XML to Entities that execute the regulations and to a Governance Spider. The exposure is completely public over the Internet or through private, secure connections.

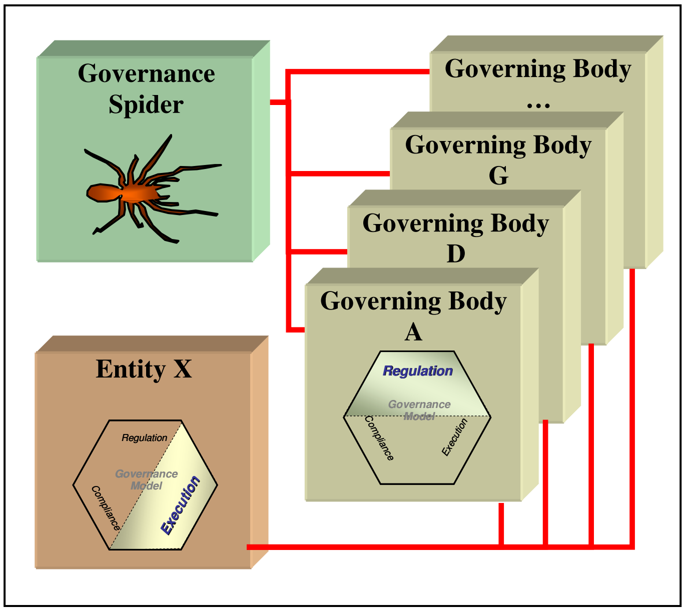


Figure 32. Governing Bodies

1.4.2 Governance Spider

The *Governance Spider* walks all Governing Body regulations registered with it and captures metadata about the Governing Body and its regulations. It places the information into a Governance Metadata data store available for query based on the governance semantics.

The Governance Spider is a semantic search engine, similar to a Google or Yahoo search engine, but aware of the semantics of governance. For example, it is possible to search for words within the Governing Statement, the Rationale or the Glossary. This makes the search efficient for the end user (human or automated process). This opens the possibility to formulate searches for all Governing Statements that refer to “log” or to “publish” or for all Glossary terms that reference a particular acronym. This eliminates redundant or potentially conflicting Governing Statements and aids in the formulation of requirement sets (i.e., profiles) by Entities.

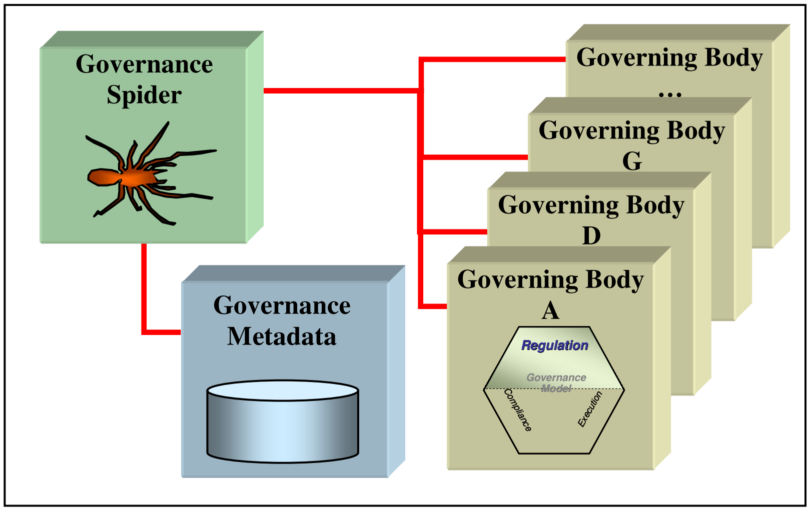


Figure 33. Automatic Capture of Governance Metadata

1.4.3 Governance Metadata Repository

The *Governance Metadata Repository* contains data about governance data maintained by Governing Bodies. The actual authoritative data for the Governing Body is maintained by the Governing Body itself. The metadata is loaded into the Governance Metadata Repository through an open, standards based interface. However, the metadata is best collected by a Governance Spider that understands the Governance Model Semantics and can *walk* through the registered Governing Body sites to mine data. The metadata is semantically rich allowing the Governance Search Engine to formulate governance semantic based queries.

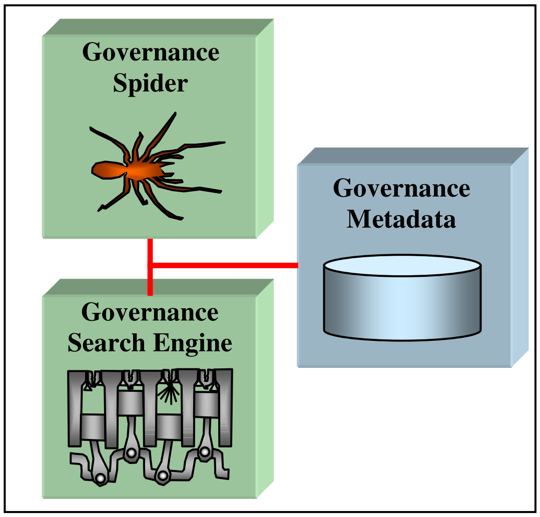


Figure 34. Governance Metadata

1.4.4 Governance Search Engine

The *Governance Search Engine* services an Entity's semantic queries about the Governing Bodies that have influence over the Entity. As with other search engines (i.e., Google, Yahoo, etc.), the search engine returns the results to the Entity. However, the results are semantic rather than syntactic in nature. For example, the results to a query made to the Governance Search Engine can be restricted to just Governing Statements, Rationale, Evaluation Criteria, Glossary or combinations thereof.

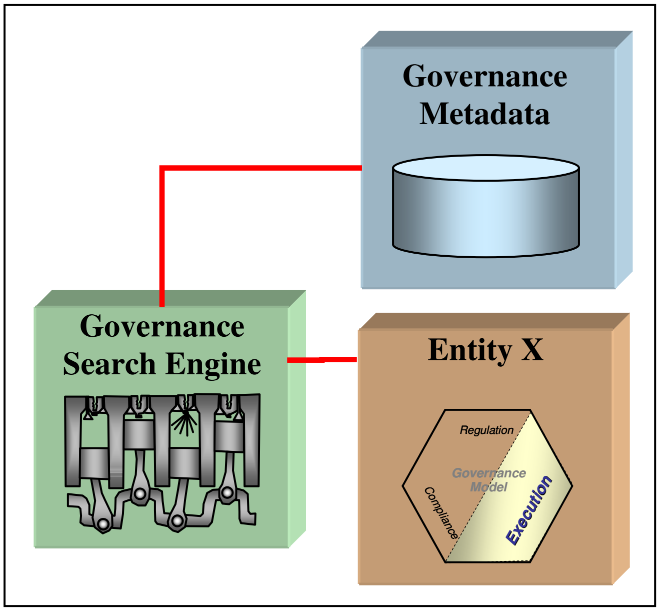


Figure 35. Searching the Governance Metadata

1.4.5 Governance Path

The *Governance Path* is a powerful concept to narrow the scope and to set the precedence of Governing Bodies that have influence over an Entity. Often an Entity is subject to any number of Governing Bodies and their regulations. The Governing Bodies that exercise influence over an Entity are not necessarily hierarchical in nature (see section 1.4.1 above).

The Entity can maintain the Governance Path locally, or maintenance may be by a standardized, formal description of the Governing Bodies, the relationship between them and the precedence of importance. External, independent tools such as a Governance Spider implementation use the Governance Path to determine how to "walk" the regulations.

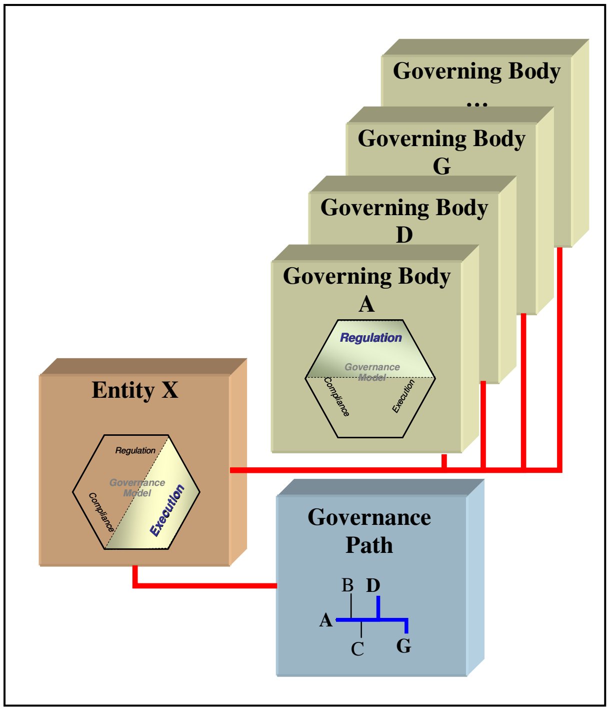


Figure 36. Setting Governing Body Scope and Precedence

References

- Ackoff, Russell L. "From Data to Wisdom," Journal of Applied Systems Analysis 16 (1989): 3-9.
- Sharma, Nikhil. *The Origin of the "Data Information Knowledge Wisdom" Hierarchy*, (1 December 2005).
- Zeleny, Milan. "Management Support Systems: Towards Integrated Knowledge Management," Human Systems Management 7, number 1 (1987): 59-70.

Appendix A– Governance Conceptual Model Rules

A.1 One Degree of Freedom Rule

One of the most important rules involves not skipping roles in the workflow inherent in the Governance Model. Each role is important and all too often, in the name of expedience, attempts are made to short circuit the model and skip roles; for example, trying to specify in Regulation Wisdom how to inspect products built during Execution. This does not, however, mean that the roles are completely isolated. Within each of the Governance Aspects, the Cognitive Model hierarchy still applies. Summarized, this the *One Degree of Freedom* rule.

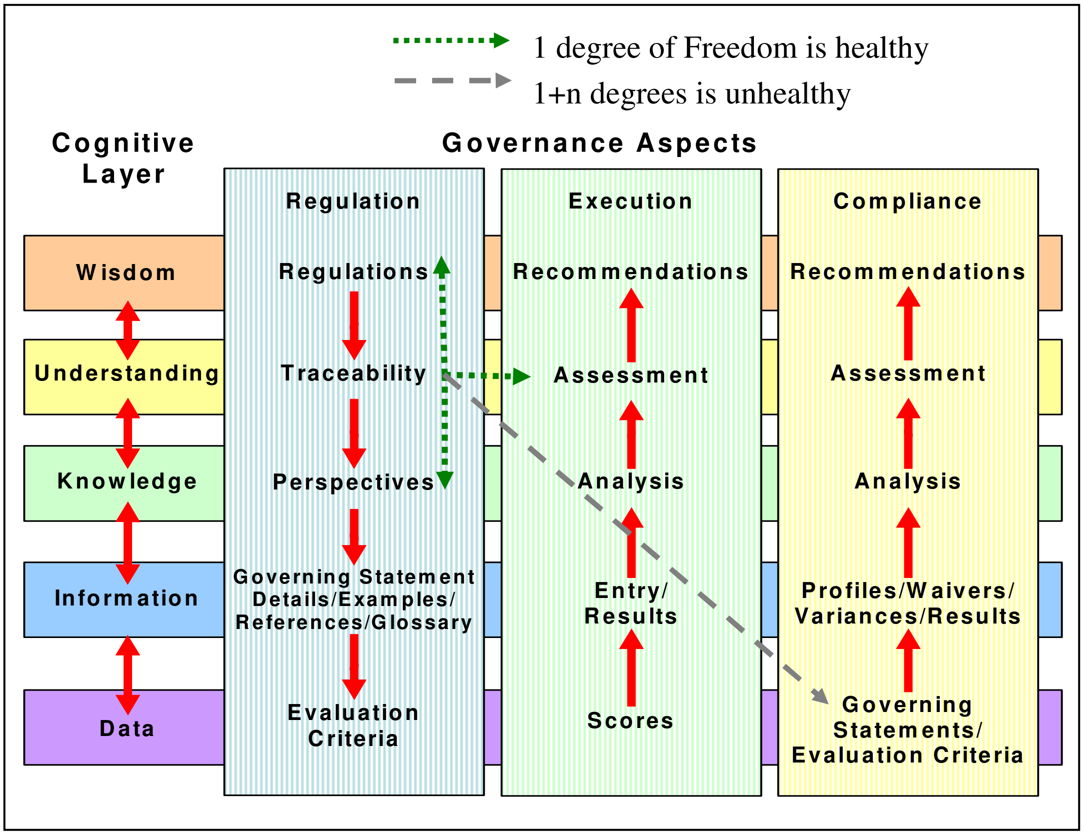
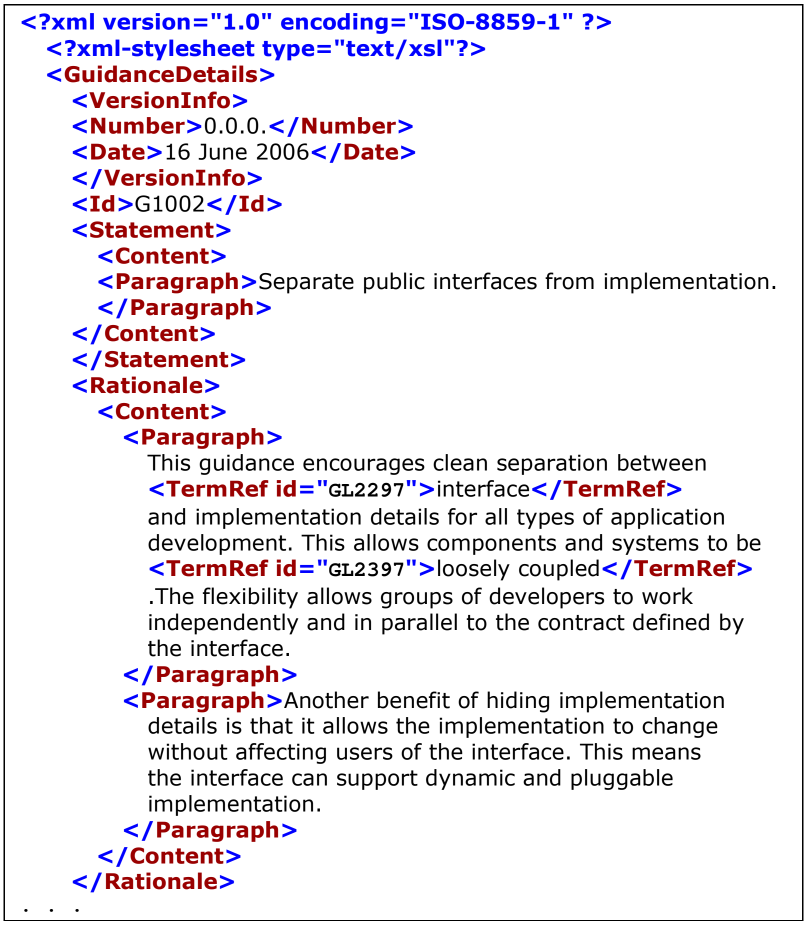


Figure 37. One Degree of Freedom Rule

A.2 Semantic Tagging Rule

Another important rule is to define the products of each Governance Model role using its own conceptual model rather than using a long, linear document that relies on format to convey meaning. This allows capturing the content semantically. The content is loaded within a relational database to facilitate access using XML and formatted according to the needs of target audiences. For example, trying to relate requirements by using the words *shall*, *will*, *should* and *may* lead to confusion and, potentially, to skipped requirements (see B.10.2).



```
<?xml version="1.0" encoding="ISO-8859-1" ?>
  <?xml-stylesheet type="text/xsl"?>
  <GuidanceDetails>
    <VersionInfo>
    <Number>0.0.0.</Number>
    <Date>16 June 2006</Date>
    </VersionInfo>
    <Id>G1002</Id>
    <Statement>
      <Content>
      <Paragraph>Separate public interfaces from implementation.
      </Paragraph>
    </Content>
    </Statement>
    <Rationale>
      <Content>
        <Paragraph>
          This guidance encourages clean separation between
          <TermRef id="GL2297">interface</TermRef>
          and implementation details for all types of application
          development. This allows components and systems to be
          <TermRef id="GL2397">loosely coupled</TermRef>
          .The flexibility allows groups of developers to work
          independently and in parallel to the contract defined by
          the interface.
        </Paragraph>
        <Paragraph>Another benefit of hiding implementation
          details is that it allows the implementation to change
          without affecting users of the interface. This means
          the interface can support dynamic and pluggable
          implementation.
        </Paragraph>
      </Content>
    </Rationale>
. . .
```

Figure 38. Example of Semantic Tagging

A.3 Nodal Networked Governing Statements

There is a tendency to think of Governing Statements as organized in hierarchical trees with one parent having multiple children and each child only having one parent. However, the organization of Governing Statements is actually a nodal network that has a few, very high level Objectives, Goals and Tenets that have any number of intermediate nodes terminated by a finite set of terminal nodes. The result is a nodal network of Governing Statements comprised of many-to-many parent-to-child and child-to-parent relationships. Some Governing Statements are Roots and often represent statutes, goals or objectives. These are the parents to any number of Intermediate Governing Statement Nodes that are refinements and interpretation of the roots. Finally, each pathway ends with at least one Terminal Governing Statement Node that has Evaluation Criteria. A Terminating Governing Statement Node can have any number of parental pathways.

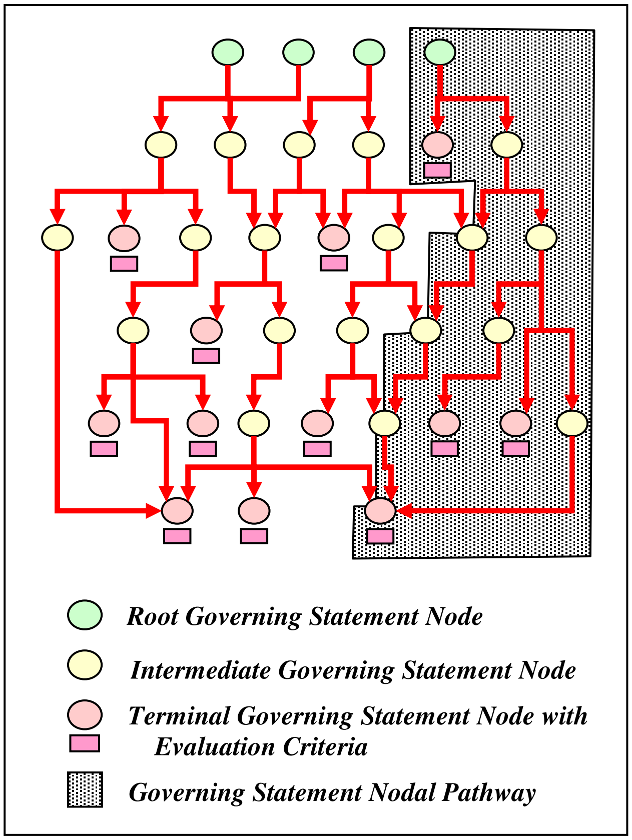

Root Governing Statement Node
Intermediate Governing Statement Node
Terminal Governing Statement Node with Evaluation Criteria
Governing Statement Nodal Pathway

Figure 39. Graphical Representation of Governing Statement Nodal Network

Appendix B – Governing Statement Traps

B.1 Non-Atomic Trap

Governing Statements are often written in prose because it is easy to string similar concepts together using lists and conjunctions. This may make for smaller documents but causes problems when interpreting the statements during the Execution and Compliance aspects of Governance. Confusion arises about which of the items in a list are required and the consequences of partially executing them. Confusion also arises from the Boolean logic levied by the use of conjunctions. The following is an example of a complex Governing Statement:

- *The network provides secure and assured transfer, storage, processing and discovery of information.*

It is not clear if the governance is to provide “secure and assured transfer,” “secure storage,” “secure processing” and “secure discovery” or if it is to provide “secure and assured transfer,” “storage,” “processing” and “discovery.” It is also not clear if the order of the items in the list indicates precedence or if there are consequences of not complying with one or more of the items. More precise forms of the Governing Statements follow:

- *The network provides assured transfer processing of information.*
- *The network provides storage processing of information.*
- *The network provides processing of information.*
- *The network provides processing discovery of information.*

B.2 Definitive Trap

When governance appears in prose, a general rule is to expand acronyms only the first time they occur within a document. This can lead to some very difficult reading of Governing Statements as it assumes that the document containing the Governing Statements will be read in a linear, top-to-bottom fashion. Governing Statements are often extracted from the original linear document format to “live” independently. Therefore, in the Governance Conceptual Model all Governing Statements have expanded acronyms. The following example contains several acronyms:

- *Network allocates IPv4 DHCP IP addresses consistent with DHCP IETF RFC 2131.*

A more complete form of the Governing Statement follows (the underlined terms indicate that hyperlinks would exist to Glossary entries):

- *Network allocates Internet Protocol Version 4 (IPv4) Dynamic Host Configuration Protocol (DHCP) Internet Protocol (IP) addresses consistent with DHCP Internet Engineering Task Force (IETF) Request for Comment (RFC) 2131.*

Another form of the Definitive Trap is the use of terms which do not have the common English definition for words or terms in the Governing Statement as in the following example:

- *All storage devices can manage residue.*

The definition for the term *residue* in this example is "Data left in storage after information processing operations are complete, but before degaussing or overwriting has taken place." This definition differs from the common English definition; thus, the governance definition needs to be in a separate Glossary with a reference (usually a hyperlink as indicated below) to this definition.

- *All storage devices can manage residue.*

B.3 Succinctness Trap

Sometimes the author of a Governing Statement feels the need to elaborate or expound on the meaning of the statement. Consequently, the Governing Statement has a tendency to ramble and violate the succinctness part of the definition of a Governing Statement. For example, the following is a Succinctness Trap example of a Governing Statement provided in a Real-Time Guidance Document:

- *In some limited cases, the data marshalling mechanisms provided by the standards based middleware may not be sufficient in terms of functionality or performance. In some very limited circumstances, such as for interfaces with legacy systems, standards-based middleware may not be employed. In those instances, care should be taken to properly align data to minimize the processing required to achieve data marshalling.*

Much of the guidance offered by this Governing Statement is rationale for why the guidance is provided. Though this information can be useful, obfuscating the actual governance with the rationale is confusing. This Governing Statement is actually trying to state the following:

- *Align data on even word boundaries.*

B.4 Implementation Trap

Implementation Traps occur when Governing Statements go beyond trying to capture "what" needs to be done and starts to capture "how" to do it. This introduces risk by the author of the Governing Statement because the "how" provided within the governing statement has the potential to be more expensive then other solutions, especially those that leverage new technology. The following is an example of an Implementation Trap:

- *Network management systems shall assign public routable IPv4 addresses using address blocks managed by the IT Department.*

The use of the word *systems* implies that the implementation requires a system rather than a service, application, or a collection of services or applications. The Governing Statement also assumes there is an IT Department, implying an organizational structure and responsibilities that may change independently of the need to assign blocks of IP addresses. The Governing Statement is more effective when transformed into the following:

- *Public routable Internet Protocol Version 4 (IPv4) addresses are assignable using address blocks.*

B.5 Operation Trap

Operation Traps occur when Governing Statements go beyond trying to capture "what" the system needs to do and start to capture "why" it will be done. This creates Governing Statement clutter that provides definitions in the guise of Governing Statements (i.e., requirements). The following is an example of an Operation Trap:

- *Comply with industry open standards to promote interoperability, agility, and long-term technical evolution.*

The expression "to promote interoperability, agility, and long-term technical evolution" provides the "why." A more precise form of the Governing Statement follows:

- *Computing Infrastructure (CI) complies with open standards defined in the DoD Information Technology Standards Registry (DISR).*

B.6 Confusing Term Trap

There are a number of terms that can confuse the purpose of the Governing Statement and often result in costly requirements creep. Here is a partial list of confusing terms:

- *Support*
- *But not limited to*
- *Etc.*
- *And/Or*

B.6.1 "Support"

The term *support* is ambiguous leaving the Governing Statement open ended, non-deterministic, and very subjective, which can lead to conflicts between the supplier and beneficiary of the support. The following is an example use of the term:

- *The system shall support dynamic configuration, rapid deployment, and provisioning for end-to-end services, with fault detection and situational awareness.*

A more concise form of the Governing Statement follows:

- *The system has dynamic configuration of end-to-end system services.*

B.6.2 "But not limited to"

Often authors of Governing Statements avoid being specific by using the "but not limited" to expression. This is an open-ended statement that provides no guidance other than the specific items listed in the Governing Statement and can be a source of contention during execution as in the following example:

- *Networks can identify and authenticate entities, including but not limited to users, networks, devices, and end systems.*

This statement not only uses the confusing "but not limited to" term but is also non-atomic. New Governing Statements are necessary as new "entities" evolve. More concise Governing Statements follow:

- *Networks identify users.*

- *Networks identify networks.*
- *Networks identify devices.*
- *Networks identify system end points.*
- *Networks authenticate users.*
- *Networks authenticate networks.*
- *Networks authenticate devices.*
- *Networks authenticate system end points.*

B.6.3 "Etc."

The use of *et cetera* (usually in the abbreviated form, etc.) is very open ended especially if the list is not preceded by a "for example" or "e.g." term. Many times, the items in the et cetera list should be part of a Glossary term that precedes it. The following is an example of the confusing, open ended "etc." term:

- *Networks shall be capable of rapidly configuring and integrating new systems (components, products, services, etc) as they become available to the network.*

This governing Statement attempts to define what new systems are as well as leaving the definition open ended. The Governing Statement is less open ended by replacing the reference to the generic expression *system* and its implied definition with a more specific term *network component* which is defined externally in a glossary. The glossary definition can evolve independently of the Governing Statement. This requires independent versioning of Governing Statements and Glossary items whenever a Governing Body or Community of Interest mandates a Governing Statement. A more concise form of the Governing Statement follows:

- *The network adapts when network components change.*

B.6.4 "And/Or"

The use of the *and/or* expression introduces non-deterministic Boolean logic errors to a Governing Statement. It is not clear which of the items in the and/or list is required, if any. By creating a separate statement for any *or* items, the business case logic determines if the items are required or not. In addition, the *and* and the *or* terms usually violate the atomicity rule. The following is an example of the non-deterministic *and/or* term:

- *The network shall be capable of dynamically supporting all network users, including those transitioning across operational and network domains and/or Community of Interest (COI) boundaries.*

This is confusing because the Boolean logic is hard to follow. Is the Governing Statement specifying transitioning across all operational, network and COI boundaries or transitioning between "operational and network boundaries" or "COI Boundaries"? More precise forms of the Governing Statement follow:

- *The network users can transition across operational domains.*
- *The network users can transition across network domains.*
- *The network users can transition across Communities of Interest (COI).*

B.7 Ambiguous Term Trap

A major cause of unverifiable requirements is the use of ambiguous terms that are subjective and often add little to the meaning. For example, no one intentionally designs systems, products, applications or services that are insufficient, user hostile or slow. Governing Statements that specify minimizing or maximizing various qualities without providing thresholds are open ended because there is no clear point at which the minimizing or maximizing is complete. Avoid ambiguous term traps by providing a list of terms to avoid. The following lists some ambiguous words:

- *Minimize*
- *Maximize*
- *Efficient*
- *Rapid*
- *User-friendly*
- *Easy*
- *Sufficient*
- *Adequate*
- *Quick*

The following is an example of a Governing Statement that uses several of the ambiguous terms:

- *The network end-to-end mechanisms shall be rapid, efficient and minimize the use of intervening translation devices.*

To create a more concise form of the Governing Statement, the ambiguous terms need explicit definitions.

- *The network uses no more than three intervening translation devices in any end-to-end path.*

B.8 Over Specification Trap

Over specification occurs when the line between regulation and execution is blurred. It is the role and the responsibility of a regulation's Governing Statements to specify *what* needs to be done and it is the responsibility of the execution's program, project or initiative to determine *how* and *where*. For example, a regulation's Governing Statement may be "Use Open Standards." During the execution of the Governing Statement by a project, the project manager decides to use networks, Web services, applications, systems and databases. If the regulation and the execution are combined, the following over specified Governing Statements result:

- *Networks use open standards.*
- *Web services use open standards.*
- *Applications use open standards.*
- *Systems use open standards.*

- *Databases use open standards.*

Instead, create a single Governing Statement with a unique identifier:

- *Use open standards. [GS1000]*

Then, during the execution of the Governing Statement by a project, program or initiative, the requirements simply refer to the Governing Statement identifiers:

- *The network shall comply with the following Governing Statements: GS1000, GS1266 and GS3045.*
- *The Web services shall comply with the following Governing Statements: GS1000, GS1123 and GS4000.*
- *The applications shall comply with the following Governing Statements: GS1000, GS1287 and GS6452.*
- *The systems shall comply with the following Governing Statements: GS1000, GS1123 and GS4321.*
- *The databases shall comply with the following Governing Statements: GS1000, GS1234 and GS253.*

B.9 Passive Voice Trap

Governing Statements delineate actions; use active voice for verbs to indicate clearly the necessary action. Passive voice leaves the identity of the actor performing the action (i.e., verb) undefined. Using forms of the verb *to be* frequently identify passive voice.

This is an example of a Governing Statement using passive voice:

- *A Directory Service is provided that all components can use. [GS1001]*

Changing this from passive to active voice clarifies the action inherent in this statement:

- *Provide a Directory Service that all components can use. [GS1001]*

B.10 Additional Governing Statement Concerns

B.10.1 Governing Statement Identifiers

Each Governing Body creates, tracks, and maintains unique Governing Statement identifiers. The identifiers do not capture any functionality, structure or organization. Often identifiers attempt to use a *hash* scheme as an aid to human memory. However, this implies that the identifiers have inherent organization and structure that is static and hierarchical. In the following example, a theoretical Net-Centric Governing Body (NCENT) has a network component (NETW) that has separate regulations for routing (ROUT) and management (MGMT).

- *Record all exceptions. [NCENT-NETW-ROUT-0010]*
- *Dynamically allocate Internet Protocol (IP) addresses. [NCENT-NETW-MGMT-0010]*

Unfortunately, the organization of the Governing Statements can change through time. The "Record all exceptions" is a good general regulation and can be promoted as applicable for every aspect covered by the Governing Body. The "Dynamically allocate Internet Protocol (IP) addresses" could move from a management Governing Statement or potentially both management and routing. A more appropriate way of numbering the Governing Statements follows:

- *Record all exceptions. [GS0010]*
- *Dynamically allocate Internet Protocol (IP) addresses. [GS0011]*

B.10.2 Use of Shall, Will, Should, Must

The use of *Shall*, *Will*, *Should* and *Must* embedded in body of prose documents often is an attempt to differentiate descriptive information, requirements, facts and goals statements. As a general rule, the following apply:

- *Requirements statements contain the word shall*
- *Statements of fact contain the word will*
- *Goal statements use the word should*
- *All other statements are descriptive*

However, Governing Statements are part of the Regulation aspect of Governance and the determination of the applicability to any particular effort is part of the Execution aspect of Governance. Generally, applicability is a business decision which can vary across the spectrum of execution implementations. Many problems occur when the Regulation and Execution aspects of Governance are intermixed. The following example mixes Regulation and Execution into a single Governing Statement:

- *Service functionality shall be exposed to the Internet using a registry.*

It may be a poor business decision to expose all services on the Internet. It is more realistic to assume, for example, that there will be one set of services exposed to the Internet, a second set of services exposed to an intranet or the Internet, and a third set of services confined to an intranet. A more concise approach would be to create a Governing Statement controlled by the Governing Body and to create a requirement statement for each execution implementation. The Governing Statement would look like this:

- *Expose service functionality to the internet using a registry. [GS1234]*

The requirements statement for each of the tiers would look like this:

- *Tier 1 shall adhere to GS1234.*
- *Tier 2 should adhere to GS1234.*
- *Tier 3 shall not adhere to GS1234.*

Appendix C – Sample XML Schema Definitions

This appendix contains examples of Engineering Governance XML Schema Definition (XSD) files used by the XML implementation of the Net-Centric Enterprise Solutions for Interoperability (NESI) project. For additional information, please see the NESI Public Site, http://nesipublic.spawar.navy.mil; NESI Parts 3 – 6 are available for viewing online using NESI-X or as Portable Document Files created from NESI-X reports.

The NESI-X implementation uses the content.xsd example below to provide semantically consistent markup of general free-form text. The current NESI-X implementation includes XSDs for Governance Details (Guidance and Best Practices in NESI-X), Perspectives, Glossary Items, References, and Audio Visual Items (Images). NESI-X presently does not implement Compliance or Execution; therefore, there are no XML Schema Definition examples in this appendix for these two aspects of the Governance Model.

C.1 Content.xsd

```
<?xml version="1.0"?>
<!-- $Revision: 1.1.1.1 $ -->
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns="http://nesi.spawar.navy.mil/nesix"
   targetNamespace="http://nesi.spawar.navy.mil/nesix"
   elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xsd:element name="Content" type="AnyContentType">
    <xsd:annotation>
      <xsd:documentation>Defines NESI-X-ML Content tags</xsd:documentation>
    </xsd:annotation>
  </xsd:element>
  <xsd:element name="OrderedList" type="ListType"/>
  <xsd:element name="UnorderedList" type="ListType"/>

  <xsd:complexType name="AnyContentType" mixed="true">
    <xsd:choice minOccurs="0" maxOccurs="unbounded">
      <xsd:element ref="Content" minOccurs="0" maxOccurs="1"/>
      <xsd:element name="Paragraph" type="AnyContentType"/>
      <xsd:element name="TermRef" type="GlossaryReferenceType"/>
      <xsd:element name="AcronymRef" type="GlossaryReferenceType"/>
      <xsd:element name="GuidanceRef" type="xsd:string"/>
      <xsd:element name="ReferenceRef" type="xsd:string"/>
      <xsd:element name="PerspectiveRef" type="PerspectiveReferenceType"/>
      <xsd:element ref="UnorderedList"/>
      <xsd:element ref="OrderedList"/>
      <xsd:element name="ExternalRef" type="ReferenceType"/>
      <xsd:element name="Image" type="ImageType"/>
      <xsd:element name="Table" type="TableType"/>
      <xsd:element name="DefinitionList" type="DefinitionListType"/>
      <xsd:element name="Note" type="AnyContentType"/>
      <xsd:element name="SubSection" type="SubSectionType"/>
      <xsd:element name="Block" type="xsd:string"/>
      <xsd:element name="Code" type="AnyContentType"/>
      <xsd:element name="InLineCode" type="xsd:string"/>
      <xsd:element name="BookTitle" type="AnyContentType"/>
      <xsd:element name="Action" type="xsd:string"/>
      <xsd:element name="VariableText" type="xsd:string"/>
      <xsd:element name="FileName" type="xsd:string"/>
      <xsd:element name="Subscript" type="AnyContentType"/>
      <xsd:element name="Superscript" type="AnyContentType"/>
      <xsd:element name="Emphasis" type="AnyContentType"/>
      <xsd:element name="Space"/>
      <xsd:element name="Break"/>
      <xsd:element name="Amp"/>
      <xsd:element name="Lt"/>
      <xsd:element name="Gt"/>
    </xsd:choice>
  </xsd:complexType>
  <xsd:complexType name="ListType">
    <xsd:choice minOccurs="0" maxOccurs="unbounded">
      <xsd:element name="Item" type="AnyContentType"/>
      <xsd:element ref="UnorderedList"/>
      <xsd:element ref="OrderedList"/>
    </xsd:choice>
  </xsd:complexType>
  <xsd:complexType name="ReferenceType" mixed="true">
    <xsd:attribute name="Url" use="required"/>
  </xsd:complexType>
  <xsd:complexType name="ImageType" mixed="true">
    <xsd:attribute name="Height" use="optional"/>
    <xsd:attribute name="Width" use="optional"/>
    <xsd:attribute name="id" type="xsd:string" use="required"/>
  </xsd:complexType>
  <xsd:complexType name="TableType">
    <xsd:sequence>
      <xsd:element name="HeaderRow" type="RowType" minOccurs="0"
         maxOccurs="unbounded"/>
      <xsd:element name="Row" type="RowType" minOccurs="0"
         maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="RowType">
    <xsd:choice>
      <xsd:element name="Cell" type="AnyContentType"
        minOccurs="0" maxOccurs="unbounded"/>
    </xsd:choice>
  </xsd:complexType>
  <xsd:complexType name="DefinitionListType">
    <xsd:sequence>
      <xsd:element name="Definition" type="DefinitionType" minOccurs="0"
        maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="DefinitionType">
    <xsd:sequence>
      <xsd:element name="TermRef" type="AnyContentType"/>
      <xsd:element name="Meaning" type="AnyContentType"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="SubSectionType">
    <xsd:sequence>
      <xsd:element name="Title" type="xsd:string"/>
      <xsd:element name="Body" type="AnyContentType"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="GlossaryReferenceType" mixed="true">
    <xsd:attribute name="id" type="xsd:string"/>
  </xsd:complexType>

  <xsd:complexType name="PerspectiveReferenceType" mixed="true">
    <xsd:attribute name="id" type="xsd:string"/>
  </xsd:complexType>
</xsd:schema>
```

C.2 Regulation Aspect Examples

C.2.1 Guidance.xsd

```
<?xml version="1.0"?>
<!-- $Revision: 1.3 $ -->
<xsd:schema xmlns:xsd=http://www.w3.org/2001/XMLSchema
      xmlns="http://nesi.spawar.navy.mil/nesix"
      targetNamespace=http://nesi.spawar.navy.mil/nesix
      elementFormDefault="qualified"
      attributeFormDefault="unqualified">

  <xsd:include schemaLocation="Content.xsd"/>
  <xsd:include schemaLocation="History.xsd"/>
  <xsd:element name="Guidance">
    <xsd:annotation>
      <xsd:documentation>
         Defines a details object (Guidance or Best Practice)
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="GuidanceId" minOccurs="1" maxOccurs="1"/>
        <xsd:element name="State" minOccurs="0" type="xsd:string"/>
        <xsd:element name="Revision" minOccurs="1" maxOccurs="1"
         type="xsd:integer"/>

        <xsd:element name="Statement" minOccurs="0" maxOccurs="1">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element ref="Content"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>

        <xsd:element name="Rationale" minOccurs="0" maxOccurs="1">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element ref="Content"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>

        <xsd:element name="Justifies" minOccurs="0" maxOccurs="1">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element ref="GuidanceId" minOccurs="0"
                maxOccurs="unbounded"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>

        <xsd:element name="DerivedFrom" minOccurs="0" maxOccurs="1">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element ref="GuidanceId" minOccurs="0"
               maxOccurs="unbounded"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>

        <xsd:element name="ReferencedBy" minOccurs="0" maxOccurs="1">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="PerspectiveRef" type="PerspectiveRefType"
               minOccurs="0" maxOccurs="unbounded"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>

        <xsd:element name="AcquisitionPhase" type="xsd:string"
          minOccurs="0" maxOccurs="1"/>

        <xsd:element name="AuthorizedBy" type="xsd:string"
          minOccurs="0" maxOccurs="1"/>

        <xsd:element name="Evaluation" minOccurs="0" maxOccurs="1">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="TestItem" type="TestItemType"
                minOccurs="0" maxOccurs="unbounded"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="GuidanceId" type="xsd:string"/>
  <xsd:element name="Tag">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="TagItem" maxOccurs="unbounded">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="TagName" type="xsd:string"/>
              <xsd:element name="TagOwner" type="xsd:string"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:complexType name="PerspectiveRefType">
    <xsd:sequence>
      <xsd:element name="PerspectiveId" type="xsd:string"/>
      <xsd:element name="PerspectiveName" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="TestItemType">
    <xsd:sequence>
      <xsd:element name="TestNumber" type="xsd:integer"/>
      <xsd:element name="Test" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element ref="Content"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="Procedure" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element ref="Content"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="Example" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element ref="Content" minOccurs="0" maxOccurs="1"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element ref="Tag" minOccurs="0" maxOccurs="1"/>
      <xsd:element name="TestItem" type="TestItemType"
        minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

C.2.2 Perspective.xsd

```
<?xml version="1.0"?>
<!-- $Revision: 1.2 $ -->
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns="http://nesi.spawar.navy.mil/nesix"
      targetNamespace="http://nesi.spawar.navy.mil/nesix"
      elementFormDefault="qualified"
      attributeFormDefault="unqualified">

  <xsd:include schemaLocation="Content.xsd"/>
  <xsd:include schemaLocation="Reference.xsd"/>

  <xsd:element name="Perspective">
    <xsd:annotation>
      <xsd:documentation>Defines a perspective</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>

        <xsd:element name="PerspectiveId" type="xsd:string"/>
        <xsd:element name="State" minOccurs="0" maxOccurs="1"
          type="xsd:string"/>
        <xsd:element name="Revision" minOccurs="1" maxOccurs="1"
           type="xsd:integer"/>
        <xsd:element name="Name" minOccurs="0" maxOccurs="1"
          type="xsd:string"/>

        <xsd:element name="Overview">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element ref="Content" minOccurs="0" maxOccurs="1"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>

        <xsd:element name="Example" minOccurs="0" maxOccurs="1">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element ref="Content" minOccurs="0" maxOccurs="1"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>

        <xsd:element name="GuidanceList" minOccurs="0" maxOccurs="1">
          <xsd:complexType>
            <xsd:sequence>
```

```
              <xsd:element name="GuidanceRef"
                     minOccurs="0"
                     maxOccurs="unbounded">

                <xsd:complexType>
                  <xsd:sequence>

                    <xsd:element name="GuidanceId"
                           type="xsd:string"/>

                    <xsd:element name="Statement">
                      <xsd:complexType>
                        <xsd:sequence>
                          <xsd:element ref="Content"/>
                        </xsd:sequence>
                      </xsd:complexType>
                    </xsd:element>

                  </xsd:sequence>
                </xsd:complexType>

              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>

        <xsd:element name="References" minOccurs="0" maxOccurs="1">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element ref="Reference"
                     minOccurs="0"
                     maxOccurs="unbounded"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

C.2.3 GlossaryItem.xsd

```
<?xml version="1.0"?>
<!-- $Revision: 1.2 $ -->
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
     xmlns="http://nesi.spawar.navy.mil/nesix"
     targetNamespace="http://nesi.spawar.navy.mil/nesix"
     elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xsd:include schemaLocation="Content.xsd"/>
  <xsd:element name="GlossaryItem">
    <xsd:annotation>
      <xsd:documentation>Defines a glossary item</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="GlossaryId" type="xsd:string"/>
        <xsd:element name="State" minOccurs="0" maxOccurs="1"
          type="xsd:string"/>
        <xsd:element name="Revision" minOccurs="1" maxOccurs="1"
          type="xsd:integer"/>
        <xsd:element name="Term" minOccurs="0" maxOccurs="1"
          type="xsd:string"/>
        <xsd:element name="Definition">
```

```
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element ref="Content" minOccurs="0" maxOccurs="1"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="Acronym" type="xsd:string" minOccurs="0"
          maxOccurs="1"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

C.2.4 Reference.xsd

```
<?xml version="1.0"?>
<!-- $Revision: 1.2 $ -->
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns="http://nesi.spawar.navy.mil/nesix"
   targetNamespace="http://nesi.spawar.navy.mil/nesix"
   elementFormDefault="qualified"
   attributeFormDefault="unqualified">
  <xsd:include schemaLocation="Content.xsd"/>
  <xsd:element name="Reference">
    <xsd:annotation>
      <xsd:documentation>Defines a glossary item</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="ReferenceId" type="xsd:string"/>
        <xsd:element name="State" minOccurs="0" maxOccurs="1"
          type="xsd:string"/>
        <xsd:element name="Revision" minOccurs="1" maxOccurs="1"
          type="xsd:integer"/>
        <xsd:element name="ReferenceKind" minOccurs="0"
          maxOccurs="1" type="xsd:string"/>
        <xsd:element name="ReferenceName" minOccurs="0"
          maxOccurs="1" type="xsd:string"/>
        <xsd:element name="ReferenceText" minOccurs="0"
            maxOccurs="1">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element ref="Content" minOccurs="0"
                maxOccurs="1"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>

        <xsd:element name="ReferencedBy" minOccurs="0" maxOccurs="1">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="PerspectiveRef" type="PerspectiveRefType"
                minOccurs="0" maxOccurs="unbounded"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

C.2.5 AudioVisualItem.xsd

```
<?xml version="1.0"?>
<!-- $Revision: 1.4 $ -->
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://nesi.spawar.navy.mil/nesix"
targetNamespace="http://nesi.spawar.navy.mil/nesix"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xsd:include schemaLocation="Content.xsd"/>
  <xsd:element name="AudioVisualItem">
    <xsd:annotation>
      <xsd:documentation>
        Defines an audio visual item such as an image
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="AudioVisualId" type="xsd:string"/>
        <xsd:element name="State" minOccurs="0" maxOccurs="1"
          type="xsd:string"/>
        <xsd:element name="Revision" minOccurs="1"
          maxOccurs="1" type="xsd:integer"/>
        <xsd:element name="Name" minOccurs="0" maxOccurs="1"
          type="xsd:string"/>
        <xsd:element name="Caption" minOccurs="0" maxOccurs="1"
          type="xsd:string"></xsd:element>
        <xsd:element name="Filename" type="xsd:string"/>
        <xsd:element name="Data" minOccurs="0" type="xsd:string"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```